Android Bluetooth Electrocardiogram

A personalized home monitoring application for heart health

Michael Atkinson Patrick Cousineau James Hollinger Chris Rennie Brian Richter
Supervised by Dr. Jens Weber

Table of Contents

Table of Figures

1.0 Summary

We set out to design and build a low cost electrocardiogram device which can be easily used by someone with low technical expertise for around-the-clock ECG self-monitoring and feedback. The project consists of the hardware and software for a wireless electrocardiogram device, and a corresponding Android application which, together, displays and records the user's heart signal. The specific goals were motivated by several 399 projects from fall 2012 (one detailing the hardware and another detailing the Android application) but were soon trimmed down due to time constraints and the broad expertise required. The prototype we delivered was able to: read the voltage signal generated by the test subject, filter out the background noise, amplify the signal to a manageable level, convert this analog signal to 10 digital bits, transmit this signal over Bluetooth, display the data graphically on the paired Android device, and save the data for future reference within the application. We feel that this prototype could be valuable as a reference or starting point for teams looking to either construct a similar project or to extend an existing product to meet a goal detailed in Section 6.0.

2.0 Problem Description

This project was based on several 399 projects from fall 2012 - one detailed the hardware [1] , and another outlined the software specifics [2] for this device. The hardware project outlined a system with active movement artefact correction along with a filter and amplification system. The adaptive filter was deemed ineffective early in this project, and considered as a possible future improvement, with the main goal being to build the amplification and filter system which communicates with an Android smartphone application. At its core, the 399 software project outlined an application that was constantly running in the background, reading ECG information over Bluetooth, giving feedback to the user, and saving the data. We decided the “user feedback” would be a real-time chart of the signal due to its intuitiveness and accessibility.

The device will not offer any medical opinions, but could be valuable in monitoring for non-continuous heart conditions and heart condition triggers that are not replicated in a hospital or clinical setting. An example application is for monitoring an elderly patient with infrequent cardiac issues. As a result of the targets, the user would not need to continuously monitor the device, but could save and send abnormal conditions as they are encountered, with a warning potentially being sent if the patient experiences abnormal conditions.

2.1 Currently Existing Alternatives

2.2 Electrocardiograms

Electrocardiograms (ECGs) are used by medical professionals to monitor the heart of a patient. These devices usually operate with up to 12 leads connected to the patient’s skin in a prescribed pattern. An ECG can be used to detect abnormal cardiovascular symptoms, measure heart rates, and monitor heart diseases. The most common non-medical application of an ECG is to measure a heart rate during a workout; however, the aim of this project is to prototype a device that could aid remote monitoring and feedback.

In addition to hospital based systems, there are also long-term home monitoring systems such as Holter monitors. These systems record 3 to 12 electrodes worth of data onto the device, and are then brought in by the patient for analysis. These monitors are intended to be used over longer periods or to test for off-site conditions such as daily routine or specific triggers. [3]

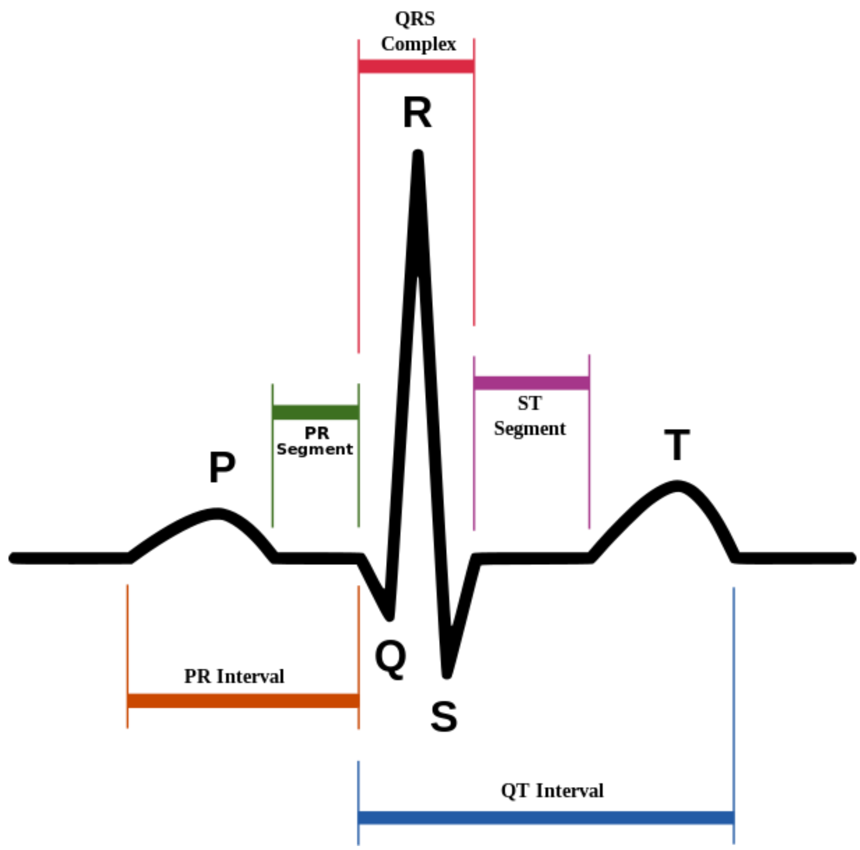


Figure 1 - Typical Heart Signal Waveform

The most important components of the ECG waveform that the device needs to report include the P wave, QRS interval, and the T wave. These waveforms have the inherent issue that the measurable signals have amplitudes in the range of 0.1mV to 1mV on top of large DC signals. Another issue with these signals is that the smallest time components last as little as 50ms (The PR segment), or 80ms (The entire QRS complex). This short time means that the sampling speed for the signal needs to be significantly less than 25ms to ensure adequate sampling. The diagram below shows the important components of the ECG waveform. [4]

2.3 Wireless ECG Systems

There are some commercially available ECGs, although the ones analyzed for this project were application specific, for example, Nike offers several wireless systems that transmit to either an Apple product or similarly enabled exercise equipment. These closed systems are not available for active reporting, and can cost over $70 for the hardware components. The goal of this project was to create data that can be saved and used for any potential activity, and shared as the user wishes.

3.0 Hardware - building an ECG

The original hardware proposal was to build the circuit using a series of amplifiers and filters. Appendix C shows the full circuit. The hardware component of the project has 8 main components: electrodes, the leads, an initial amplification stage, high-pass filter, notch filter, secondary amplification stage, low-pass filter, and the analog to digital converter (ADC).

3.1 Electrodes

The final design uses a three electrode system. While a greater number of leads are commonly used in standard ECGs, three is the minimum number of electrodes that can be used to recover a signal from the heart. This design was chosen for two main reasons. First, reducing the number of leads to a minimum makes the device less intrusive and more suitable for longer term use. It would be a logistical challenge to make sure the standard 12 leads would not reduce the user's mobility yet remain attached as they went about their day, especially considering the leads attached to wrists and ankles. Second, choosing a 3 electrode system makes the hardware and software design significantly simpler and cheaper. Since only one signal is measured, only one set of amplifiers, filters and analog to digital converters are needed. As well, monitoring only one signal allows for less bandwidth and lower power when transmitting the signal by Bluetooth.

The prototype electrodes were made by soldering pennies onto leads to create a receiver for the small signal produced by the heart. Pennies were used due to their availability, low cost and high conductivity. Two of the electrodes are connected across the heart to detect a signal, while the third is placed on the body near the hip to form a ground reference. The electrodes connected across the heart are located in the V2 and V5 positions [4] as shown in Figure 2. The test subject throughout this project varied between two group members, and willing members of the public, but the final demonstration was limited to just one member to reduce the potential for damage to the system.

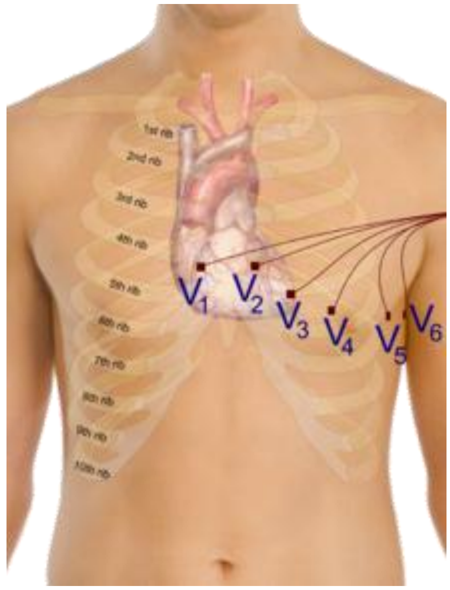


Figure 2 - Electrode Placement

3.2 Leads

A standard ECG lead is shielded to block noise from interfering with the heart signal. To decrease costs and lower complexity, the leads used in the project are unshielded wires soldered to the electrodes. Because the leads are not shielded, the 0.1-10 mV heart signal is very susceptible to noise from motion, other body functions, and background noise. In order to minimize this interference, a series of filters were employed to clean-up some of the signal. The leads used in this experiment were compared against a set of standard 3 electrode medical leads, which were owned by the department. These leads were valued at more than $130, and performed no filtering or amplification.

3.3 Isolation Amplifier

The ICs used in the circuit require a split supply of approximately ±5V to run. However, to maintain portability, it was desired to power the circuit with a 9V battery. The split supply was accomplished by using an Operational amplifier and a voltage divider to set a virtual ground. This can be seen in the figure below. The terminals of the battery now function as ±4.5V. Note that because this virtual ground is connected to the user's body through the third lead, the ground of our circuit is actually set to the base voltage of the user's body.

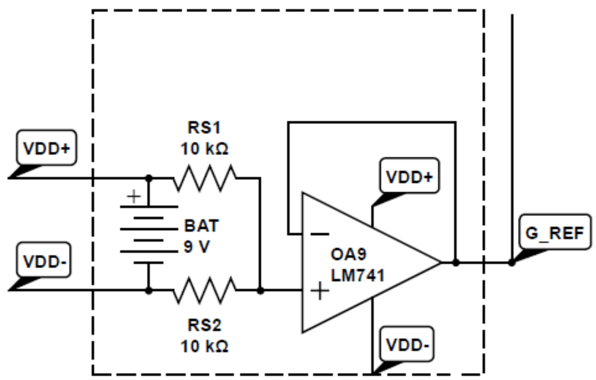


Figure 3 – Isolation and Supply Circuit

3.4 Amplifier

To recover the differential signal from the user's heart, the circuit design uses a single instrumentation amplifier with a high gain. The primary reason an instrumentation amplifier was chosen is because the heart signal has a floating reference. The model number of the chip used is the Texas Instruments INA126P. The amplifier is set to have some initial gain of 160 to increase the strength of the signal prior to filtering. However, this signal contains a large amount of noise that is also amplified. The primary sources of noise are low frequency noise from the body, high-frequency thermal and switching noise, and 60Hz power line interference due to the long leads.

3.5 High-Pass Filter

The first stage in the filter network is a passive high pass filter with a 0.5Hz corner frequency. This filter removes most of the DC offset in our signal, and the corner frequency is set very low due to our signal components residing near 1Hz. This filter is also necessary because of a baseline wander caused by the respiratory system. Without a high-pass filter the output of the ECG would be the desired 1Hz heart signal superimposed on a ~0.2Hz sine wave.

3.6 Notch Filter

The second stage is a 60Hz passive notch filter. This filter is crucial due to the unshielded leads used by the ECG. These function as antennas and pick up the 60Hz noise from power lines. This interference was the most reliable component to remove from the signal due to its predictable frequency. This filter is necessary due to the use of unshielded leads. However, it was decided that it is more cost efficient to implement a 60Hz notch filter than to use shielded leads.

3.7 Secondary Amplifier

A second amplifier composed of an Operational amplifier was added into the hardware to allow for a finer tuning of the gain. This amplifier offered a gain of about 25 times, bringing the total gain of the system to near 4000. After some attenuation in the filter stages, this would bring our estimated 1mV (peak-to-peak) signal to about 4V, with the minimum expected signal in the 0.1mV range amplified to a still detectable 400mV. As the maximum range of our ADC is 5V, amplifying the signal to near this value increases the dynamic range of our sensor.

3.8 Low-Pass Filter

The low pass filter is designed with a corner frequency of 400Hz. This removes any unwanted high frequency noise from the waveform. As well, removing unnecessary high-frequency portions of the signal will decrease signal bandwidth. A passive filter network was chosen as it is significantly easier to implement than an active filter.

3.9 Analog to Digital Converter

The ADC is a critical component of the ECG module. It takes the received ECG signal from the electrode circuitry and digitizes it for transmission to the mobile phone application. The ADC used is model MCP3008, a 10 bit ADC allowing for easy data transfer. The ADC is powered by a 5V regulator connected to the battery, with the analog ground reference connected to the 0V terminal of the battery with the reference voltage connected to the 5V supply as well. This was done to achieve a 5V range of input signal from the preceding hardware without having to account for the 2V shift that was realized by the hardware. The final ADC signal converts the estimated 2V peak-to-peak signal to a 450 bit range within the 1024 bit digital signal.

3.10 Raspberry Pi

A Raspberry Pi [5] is an ARM based, multimedia application aimed microcomputer.

The Raspberry Pi microcomputer connects the ADC and Bluetooth modules. Running a distribution of Debian Linux, the Raspberry Pi is capable of both the analog input (with the addition of the MCP3008 ADC) and digital Bluetooth transmission components of the ECG system. In the future, we would like to utilize an application-specific microcontroller for this portion of the project. For our prototyping stage, however, the Raspberry Pi is both easy to use and implement.

The ADC and the Raspberry Pi communicate using hardware SPI which is a synchronous serial protocol. It needs a clock line as well as data in and out lines. It also has a chip enable line that is used to choose which SPI device to talk to. [9]

MCP 3008 Pin		Pi GPIO Pin	
13	CLK	23	GPIO11 SPI0_SCLK
12	DOUT	21	GPIO09 SPI0_MISO
11	DIN	19	GPI010 SPI0_MOSI
10	CS	24	GPIO08 CE0
9	DGND	6	GND

Figure 4 - Raspberry Pi to ADC Connections

3.11 Bluetooth/Firmware

The selection of the Bluetooth wireless communication protocol was an easy decision. Modern Android smart phones with the ability to run the application already have a Bluetooth radio, so no extra hardware is required in order to get the communication channel up and running. In addition, the ECG component would be simple to pair using this technology; extremely similar to that of a commonly utilized audio headset or speaker. This means that it would be very simple for the average user to get the entire ECG system up and running on their mobile device.

An important aspect to point out for this project is the overall power consumption for both the ECG hardware and the mobile device. Constantly streaming data over a wireless connection takes a large amount of power, hence why the Bluetooth 4.0 “Low Power” protocol was initiated. Unfortunately, this protocol was not implemented for the proof of concept; as neither the Android phone nor the USB radio supported it. As Bluetooth 4.0 becomes more of a standard, the availability of both the phones supporting it and the standalone transmitters will increase. As it stands, this is a strong consideration for implementation of a future aspect for the project.

At this stage of the project, we chose to utilize a Raspberry Pi Linux-based computer in order to handle the firmware side of the ECG. Utilizing a Python Script, the Raspberry Pi pairs with and connects to the phone-based Android application using a USB Bluetooth radio.

3.12 RTOS vs. non RTOS

Standard operating systems focus on doing as much computation in the shortest time where real time operating systems put an emphasis on having a predictable response times. To ensure a predictable response time in the ECG it would be ideal to be using a real time operating system (or a dedicated microcontroller). For the prototype we are building, we will be using a standard Linux distribution. In light of this, the prototype ECG, running off of a Raspberry Pi, should meet any real time constraints that we have since the processor is overpowered for our application. We believe that the overhead from running a Linux distribution should not significantly affect the sampling rate (as no other applications will be running besides our Python script).

3.13 Why not Arduino?

We decided against using a widely available programmable microcontroller (such as Arduino) because of budgeting, time, and availability issues. The group already had, in its possession, a Raspberry Pi microcomputer and a USB Bluetooth radio. In order to implement a microcontroller based solution, a standalone Bluetooth serial radio (such as a HC-05) would have been required. In the time frame of this project, we were not able to obtain this hardware component from any of our trusted suppliers. Hence, we chose to rapid prototype with the USB radio and the Pi.

4.0 Software - there's an "App" for that

4.1 Android Platform

The Android environment was chosen for ECG software for multiple reasons. One reason for choosing the Android platform is that it provides the option to run the app as a 'service', which allows it to run constantly in the background while iOS does not provide this option to developers. And, while no member had extensive mobile programming experience on either platform, Android development is Java-based; Java is a language our team was very familiar with.

4.2 Application goals

The 399 software project outlined two major purposes for the Android app. Firstly, the app must be able to analyze the ECG signal reading in real-time, and recognize any heart complications that occur or are occurring. The app should be able to analyze this in a short enough time that a "help" message can be dispatched to appropriate parties in cases of emergency. The second purpose for the app was to save the cardiogram data for a user. The motivation for this comes from the fact that the behaviour of the heart leading up to an incident is often not monitored, and could be valuable to medical professionals - questions or problems for this section include where to store the data (local, SD card, cloud service), in what format to store the data, and how to keep it confidential.

However, as the project progressed we were not able to develop a hardware prototype as quickly as we planned and thus had no device to transmit Bluetooth to test the Bluetooth capabilities. We had also made the false assumption that we had made was that Android would have a very simple Bluetooth wrapper library which would be as simple as Android's BluetoothAdapter object with which we had previous experience. Unfortunately, the Bluetooth connection involves setting up a listening Bluetooth server, and then servicing the connection - and while this is not terribly difficult, it was more than we were expecting for this segment, and the increased complexity meant more testing was required once we had a Bluetooth device to test. Bluetooth being blocked did not stop work on the whole application though: with some sample ECG data we created manually from a signal-generator we were able to work on the signal-charting component, making substantial progress.

As we put work into the software it became apparent that each of the three application goals from the project's outset was large - even the smallest task, reading the Bluetooth data, involved more effort than expected. We decided to focus on displaying the chart to the user and saving the data, and to not focus on the "user notification/signal analysis" portion. No group member is particularly well-versed in signal analysis or of ECG analysis, and testing the feature would involve simulating (or inducing) a heart attack.

5.0 Android Application

The Android application consists of multiple Java classes and several different interface screens to serve the prototype's goals of displaying real-time ECG data and saving the data. Some difficulties we encountered while creating the app were the demands on the app to display data points very quickly (after all, a real-time heart signal moves is very fast) on top of general difficulties working with a constantly-updating UI thread. Dealing with Bluetooth was another issue. The resulting prototype is robust enough to handle multiple Bluetooth disconnects or interruptions and continue charting (charting is effectively paused while no data is received). Additionally, the software was able to chart points at a rate of around one point every ten milliseconds, which was enough to produce a very detailed heart signal.

5.1 Software Components

5.1.1 MainActivity.java

MainActivity.java extends the Android Activity [6] class, and is the main activity for the application - this is an Android convention similar to the "main()" method in C programming, namely that it is the activity that is started when the app itself starts. This activity checks that the phone has the necessary Bluetooth abilities, and prompts the user to enable Bluetooth connectivity using Android Intents. This Activity also starts the Bluetooth connection Service, which is crucial for Bluetooth connectivity with the ECG. Buttons to view real-time data charting and saved data are also found on the MainActivity screen, along with our team's Bluetooth Android logo. When the user closes the app, this Activity is also responsible for saving the received data into a flat text file in the app's internal storage space on the device.

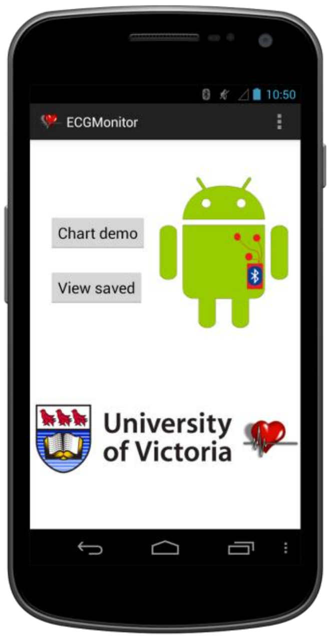


Figure 5 - MainActivity.java layout.

5.1.2 ECGChartActivity.java

This Activity is responsible for displaying the information received from the Bluetooth-receiving Service. We used the open source 3rd party Android library "achartengine" [7] to generate the graphs. The way the Android environment works is that only one Thread can update the UI - this is named the UI Thread, and is the main Thread that starts work on an Activity when an Activity starts. In our project, we required a separate Thread to constantly read Bluetooth data from an asynchronous system message queue while the UI was being updated. The communication between these Threads was accomplished using an Android Handler object, which acted as a call back onto the UI Thread. By putting the code that updated the UI into this Handler, it was guaranteed that the UI Thread would be the only Thread modifying the screen. This solved our issues of the graph crashing and allowed the chart to be updated for extended periods of time without crashing.

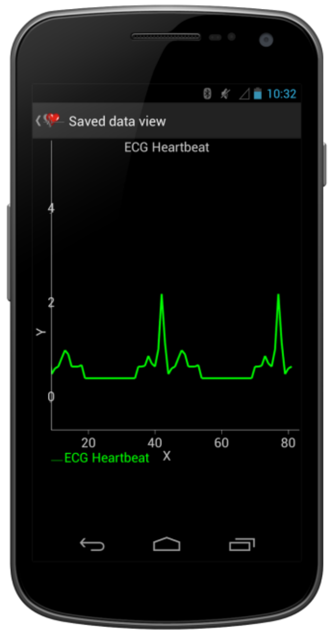


Figure 6 - Real-Time Charting Screen in Action

5.1.3 ViewSavedDataActivity.java

This Activity is very similar to the real-time charter except instead of displaying a constantly-updating chart of the data as it is received, it statically displays the entirety the data gathered since the app was started. This was also accomplished using the achartengine graphing library, and the graph is scrollable and zoomable to improve precision.

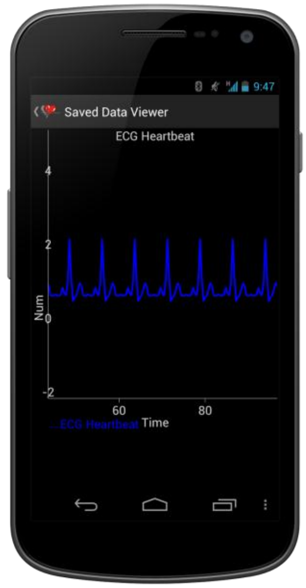


Figure 7 - Viewing All Collected Data

5.1.4 BluetoothConnectionService.java

This class extends the Android Service object, and runs in the background from when it is started until an Activity explicitly closes it, which the MainActivity does in its onDestroy (equivalent to an exit function) method. This class spawns a Bluetooth ServerThread which broadcasts the UUID and the service name of the ECG Monitor on the Bluetooth radio frequency. Any requests on that socket cause another Thread to be spawned. This AcceptThread handles the connection and the incoming stream of Bluetooth data, which is put into a system message queue.

6.0 Future Possibilities

Over the course of the project, many features and potential improvements were discussed. Some of these possibilities are listed below, but the most important and improvements to make would be to migrate to a microcontroller, use proper electrodes with optical isolation for the long term applications, and to improve the data saving capabilities of the application.

6.1 Extending the Android Application

6.1.1 Real-time heart rate monitoring and analysis

One of the main and motivating goals of this project from 399 was to examine the heart signal in real time and detect any electrical abnormalities. The app would then go on to either inform the user with a screen prompt, automatically send a message to an emergency contact, or automatically contact the nearest hospital. To accomplish the processor-heavy signal analysis required for this we recommend using Android's [8] Native Development Kit. It allows code to be written in lower-level native languages such as C, and contains fast signal processing libraries. This entire task should be tackled by a team with more experienced in signal analysis or at least a team more prepared with test signals and sample signals for verification and validation.

6.1.2 Multiple mobile devices attached to one ECG

A reasonable practical extension of this prototype is to have one patient attached to an ECG monitor and multiple stakeholders able to view or monitor the patient's heart signal from their smartphones. Our project was designed to be 1:1 between ECG device and Android application, and so the Bluetooth server portion is started from the application because it was simpler to program. To have multiple applications reading from the same device, would require the server functionality to be implemented on the firmware of the ECG, with logical to handle multiple simultaneous connections.

6.1.3 Improved data saving and analysis

Our application does indeed save the ECG data it reads, but it is not very useful in its current implementation. Because the data is currently saved to the app's designated internal storage, only the app itself can read the file. There is currently no functionality for viewing a list of all saved ECG data files in the app, and so these saved files will not be used. As the amount of data saved increases and possibly as the number of devices running this application increase, centralized data storage would be ideal. From this data warehouse, other services could be run on the entire data set, applying data mining and other analysis techniques to the data to find trends and patterns useful to medical researchers. However, once this medical information is collected in a central data area, security of the area becomes a prime concern and should be comprehensively addressed.

6.2 Firmware Improvements

6.2.1 Migration to Microcontroller

The next major iteration for the ECGs firmware would be to replace the Raspberry Pi component with a programmable microcontroller. As the firmware component's only task is to digitize and transmit the analog ECG signal, this can be easily performed on an inexpensive microcontroller (such as an Atmel ATMega328). Obviously, testing and verification of this new platform would be required in order to assure the same sampling and data rates obtained for the proof of concept. However, the monetary cost and physical size benefits would be immediate. Using an ATmega328 and HC-50, the overall hardware size could fit on one of the proto boards used for the POC circuit, completely removing the secondary ADC circuit and microcomputer components altogether. From the cost perspective, the total hardware cost for the POC comes in at around $50; whereas a final design could come in at a mere 15.

6.2.2 Low-Power Bluetooth

Another potential improvement would be to take advantage of low-power Bluetooth. While this was initially considered, the libraries for the Bluetooth module we already had made it significantly easier to implement. Additionally, the power requirements of the firmware were not taken into consideration in the prototype. This was mostly due to the use of the Raspberry Pi, which has significantly more processing ability than was needed. If a new microcontroller were used, it would be more feasible to work with low-power Bluetooth, but it is above and beyond this particular prototype.

6.3 Hardware Improvements

6.3.1 Leads

Using shielded leads would create an immediate difference in the amount of noise the circuit would experience. While the current circuit accurately removes most of the noise, better leads would reduce the amount of filtering that is necessary, and improve the accuracy of the final waveform.

6.3.2 PCB

Printing the circuit to a PCB with ground planes would reduce some noise in the system and would allow for a much smaller product. Because of how little circuit is actually involved, a printed circuit board with surface mount components would be very small.

6.3.3 Electrodes

Foam based electrodes that form to the contours of the user's body would be preferable to pennies due to better conduction. As well, electrodes that help alleviate the artefacts caused by the user's movement would be preferable. It is possible that small accelerometers in the electrodes could be used for this purpose. However, these electrode upgrades would increase the cost and complexity of the system.

6.3.4 Isolation

One necessary improvement before the ECG could be released into the public is electrical isolation. The leads are not currently isolated in any way, so it would be possible to cause damage to the circuit by electrostatic discharge across the leads. Another problem is the chance that the circuit could put a voltage across the user's heart unintentionally, causing cardiac malfunction. These problems would be addressed by using optically isolated leads, but would increase the cost of the unit.

6.3.5 Power

A better battery system would need to be developed before the hardware portion of this project could be brought to market. The current 9V battery is unnecessarily expensive and bulky, considering the size of the application. With a smaller microcontroller, a battery would be sufficient to power the entire circuit, and a battery would be developed to keep a Bluetooth system active. Ideally, this battery would be rechargeable, reducing waste.

7.0 References

7.1 Cited References

[1] Griff, D., Matzanke, K., Parry, J. and Schotlten, P. (2012, Dec) Wireless ECG Software [Website]. http://www.ece.uvic.ca/~elec399/projects_092012/www.web.uvic.ca/~mchester/software/

[2] Chester, M., Hollinger, J., Sturgess, T., Tran, J. (2012, Dec) Wireless ECG Hardware [Website]. http://www.ece.uvic.ca/~elec399/projects_092012/web.uvic.ca/~mchester/hardware/

[3] Wikipedia Contributors. (2013, Mar 26). “Holter Monitor,” *Wikipedia, the Free Encyclopedi* [Website]. http://en.wikipedia.org/wiki/Holter_monitor

[4] Wikipedia Contributors. (2013, Apr 1) “Electrocardiography,” *Wikipedia, the Free Encyclopedia* [Website]. http://en.wikipedia.org/w/index.php?title=Electrocardiography&oldid=548078993

[5] The Raspberry Pi Foundation. (2013, April 4). Raspberry Pi [Website]. Available: http://www.raspberrypi.org/

[6] Android Developers (2013, Apr 2). “Android APIs” [Website]. http://developer.android.com/reference/packages.html

[7] 4ViewSoft Company. (2012, March 30). “AChartEngine.” http://achartengine.org/

[8] Android Developers. (2013, March) “Android NDK” [Website]. https://developer.android.com/tools/sdk/ndk/index.html

[9] Stewart Russell. (2013, Feb 2). “Simple ADC with the Raspberry Pi” [Website]. http://scruss.com/blog/2013/02/02/simple-adc-with-the-raspberry-pi/

7.2 Additional References

[10] Vicente, P. F., Hoz Sevilla, R. B., Altuna, M. M. (2012, June 28). Low-Power Wireless ECG Monitoring System for Android Devices. http://eprints.ucm.es/15930/1/memoria.pdf

[11] eLinux Contributers. (2013, Apr 1). “RPi Low-level peripherals” [Website]. http://eLinux.org/RPi_Low-level_peripherals

[12] Nguyen, J. (2003, Nov). “Homemade Electrocardiograph” [Website]. http://www.eng.utah.edu/~jnguyen/ecg/ecg_index.html

Appendix A: Android Source Code

BluetoothConnService.java

```
package uvic499.software.ecg;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.util.ArrayList;
import java.util.List;
import java.util.UUID;
import java.util.concurrent.LinkedBlockingQueue;

import android.app.Service;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothServerSocket;
import android.bluetooth.BluetoothSocket;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.content.IntentFilter;
import android.os.IBinder;

public class BluetoothConnService extends Service {

  private final BluetoothAdapter BT = BluetoothAdapter.getDefaultAdapter();
  // TODO: BT board standard one is 00001101-0000-1000-8000-00805F9B34FB. Try 00001101-0000-1000-8000-
00805F9B34FA
  protected static final UUID serverUUID = UUID.fromString("00001101-0000-1000-8000-00805F9B34FA");
  private BluetoothSocket connectionSocket = null;
  private BroadcastReceiver btReceiver;
  // Intent code for Bluetooth server thread to reactivate.  String is in Android manifest xml
  private final String BLUETOOTH_ACTION_DONE_READING = "android.intent.action.BLUETOOTH_READ_DONE";

  private AcceptThread acceptThread;
  private BluetoothReadThread readThread;

  // Queue that ECGChartActivity uses to grab Bluetooth data; so far this has been fast/efficient enough
  static LinkedBlockingQueue<Double> bluetoothQueueForSaving = new LinkedBlockingQueue<Double>();
  static LinkedBlockingQueue<Double> bluetoothQueueForUI = new LinkedBlockingQueue<Double>();

  @Override
  public void onDestroy() {
    unregisterReceiver(btReceiver);
    try {
      acceptThread.cancel();
      readThread.cancel();
    } catch (Exception e) {
      // If Threads have not started, I don't know what exception will be thrown:
      // so catch generic Exception, don't crash, assume Threads are gone.
    }
    super.onDestroy();
  }

  @Override
  public IBinder onBind(Intent arg0) {
    // TODO Auto-generated method stub
    return null;
  }

  @Override
  public int onStartCommand(Intent intent, int flags, int startId) {
    System.out.println("IN BT_CONN_SERVICE's onStartCommand!");
    //addFileToQueue(); TODO: I am now trying Bluetooth stuff.  Forget dummy file writing for now
```

```
    // Create a BroadcastReceiver for BluetoothAdapter.ACTION_SCAN_MODE_CHANGED
    // This will alert us when a device disconnects so we can spawn a new AcceptThread for another
connection
    btReceiver = new BroadcastReceiver() {
      public void onReceive(Context context, Intent intent) {
        String action = intent.getAction();
        System.out.println("-----ACTION = "+action);
        // When discovery finds a device
        if (BluetoothAdapter.ACTION_SCAN_MODE_CHANGED.equals(action)) {

          // If a server thread is active, and scan mode changed, need to make sure
          // device is still discoverable
          if (acceptThread != null && acceptThread.isAlive()) {
            setDeviceToDiscoverable();
          }

        }
        else if (BLUETOOTH_ACTION_DONE_READING.equals(action)) {
          // Bluetooth channel has been disconnected.
          // Wait for readThread to finish, then start another acceptThread
          try{
            Thread.sleep(2000);
          } catch (InterruptedException e){}
          startBTServer();
        }
      }

    };

    // Now register the BroadcastReceiver to handle Bluetooth server disconnects and device scan mode
changes
    IntentFilter filter = new IntentFilter(BLUETOOTH_ACTION_DONE_READING);
    filter.addAction(BluetoothAdapter.ACTION_SCAN_MODE_CHANGED);
    registerReceiver(btReceiver, filter);

    // Start first BT server Thread
    startBTServer();

    System.out.println("--Started activity thread!");

    // We want this service to continue running until it is explicitly
    // stopped, so return sticky.
    return START_STICKY;
  }

  private void startBTServer() {
    // Always cancel device discovery - this slows BT down if it is on
    if (BT.isDiscovering()) {
      BT.cancelDiscovery();
    }

    // So that server socket is visible
    setDeviceToDiscoverable();

    // Now close the read thread if it was running
    try {
      readThread.continueReading = false;
      readThread.join();
    } catch (InterruptedException e) {
      readThread.interrupt();
    } catch (NullPointerException nulle) {
      // readThread not initialized - this is fine
    }

    //TODO: Try keeping acceptThread alive
    if (acceptThread != null && acceptThread.isAlive()) return;
    // Now close the server if it was running
    try {
      acceptThread.cancel();
```

```
    acceptThread.join();
  } catch (InterruptedException e) {
    acceptThread.interrupt();
  } catch (NullPointerException nulle) {
    // acceptThread not initialized - this is fine
  }

  acceptThread = new AcceptThread();
  acceptThread.start();
}

private void setDeviceToDiscoverable() {
  // Ensure device is discoverable so ECG can see its service.
  int scan = BT.getScanMode();
  if(scan != BluetoothAdapter.SCAN_MODE_CONNECTABLE_DISCOVERABLE) {
    Intent discoverableIntent = new Intent(BluetoothAdapter.ACTION_REQUEST_DISCOVERABLE);
    discoverableIntent.putExtra(BluetoothAdapter.EXTRA_DISCOVERABLE_DURATION, 300);
    discoverableIntent.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
    startActivity(discoverableIntent);
  }
}

class AcceptThread extends Thread {
  private final BluetoothServerSocket mmServerSocket;

  public AcceptThread() {
    // Use a temporary object that is later assigned to mmServerSocket,
    // because mmServerSocket is final
    BluetoothServerSocket tmp = null;
    try {
      // TODO: try sleeping before making a new thread so that SDP entry can be fully removed
      Thread.sleep(2000);
      // MY_UUID is the app's UUID string, also used by the client code
      tmp = BT.listenUsingRfcommWithServiceRecord("ECG Monitor", serverUUID);

    } catch (IOException e) {
      System.out.println("-- Socket listen() failed!");
    } catch (InterruptedException e){}
    mmServerSocket = tmp;
  }

  public void run() {
    System.out.println("Starting BT server...");
    // Keep listening until exception occurs or a socket is returned
    while (true) {
      try {
        connectionSocket = mmServerSocket.accept();
      } catch (IOException e) {
        System.out.println(e+"\nCouldn't mmServerSocket.accept()!");
        break;
      }

      // If a connection was accepted
      if (connectionSocket != null) {
        // Do work to manage the connection (in a separate thread for reading)

        readThread = new BluetoothReadThread();
        readThread.start();
        // Instead of closing socket, try just keeping it alive to accept multiple connections

      }
    }

    System.out.println("////// EXITED SERVER READ LOOP");
  }

  /** Will cancel the listening socket, and cause the thread to finish */
  public void cancel() {
    try {
```

```
      mmServerSocket.close();
    } catch (IOException e) { }
  }
}

class BluetoothReadThread extends Thread {

  private final InputStream iStream;
  private boolean continueReading = true;

  public BluetoothReadThread() {
    InputStream tmp = null;
    try {
      tmp = connectionSocket.getInputStream();
    } catch (IOException e) {
    }
    iStream = tmp;
  }

  @Override
  public void run() {
    List<Character> doubleBuilder = new ArrayList<Character>();
    char c;
    int waitCount=0;
    while (continueReading) {
      try {
        // Read integer values from Bluetooth stream.
        // Assemble them manually into doubles, split on newline (\n) character
        if (iStream.available() > 0) {
          waitCount = 0;
          c = (char)iStream.read();
          if (c == '\n') {
            // end of the double, put it in the queue
            char charArray[] = new char[doubleBuilder.size()];
            for (int j=0; j < charArray.length; j++) {
              charArray[j] = doubleBuilder.get(j);
            }
            String stringVal = String.copyValueOf(charArray);
            Double d = Double.valueOf(stringVal);
            System.out.println("--- queueing:" + stringVal);
            bluetoothQueueForSaving.offer(d);
            if (ECGChartActivity.isActive)
              bluetoothQueueForUI.offer(d);
            doubleBuilder.clear();
          } else {
            doubleBuilder.add(c);
          }
        } else { // No input stream available, wait
          if (waitCount >= 500000) {
            // No data ready in 500000 loop cycles, ECG has probably been disconnected. Close self.
            waitCount = 0;
            System.out.println("----wait count expired");
            continueReading = false;
            this.stopAndSendIntent();
          } else {
            waitCount++;
          }
        }

      } catch (IOException e) {
        System.out.println(e+"\nError sending data + :"+e);
        // Bluetooth error! Stop reading.
        this.stopAndSendIntent();
      }
    }
  }

  public void stopAndSendIntent() {
```

```
      this.cancel();

      Intent intent = new Intent();
      intent.setAction(BLUETOOTH_ACTION_DONE_READING);
      sendBroadcast(intent);
    }

    public void cancel() {
      System.out.println("-----Cancelling readThread!!");
      try{
        iStream.close();
      } catch (IOException e) {
      } catch (NullPointerException e){};

      continueReading = false;
    }
  }
  /*
   * TODO: This method is probably obsolete now
   *
   * Method to add data to the Bluetooth queue - for now, loop through a file because
   * we don't have BT connection stuff.
   * Read file in new Thread, add doubles to queue.
   */
  public void addFileToQueue() {

    Thread t = new Thread(new Runnable() {
      public void run() {
        // Loop through the test data to get more heartbeats
        while (true) {
          if (bluetoothQueueForSaving.size() > 2000) {
            try{
              Thread.sleep(3000);
            } catch (InterruptedException e) {
              continue;
            }
          }
          readFile(R.raw.short_ecg);
        }
      }
    });
    t.start();
  }

  // Helper method for adding hard-coded ecg file data to Bluetooth queue
  private void readFile(int resourceId) {
    // The InputStream opens the resourceId and sends it to the buffer
    InputStream is = this.getResources().openRawResource(resourceId);
    BufferedReader br = new BufferedReader(new InputStreamReader(is));
    String readLine = null;

    try {
      // While the BufferedReader readLine is not null
      while ((readLine = br.readLine()) != null) {
        bluetoothQueueForSaving.offer(Double.valueOf(readLine));
      }

    // Close the InputStream and BufferedReader
    is.close();
    br.close();

    System.out.println("Done reading file!\n");

    } catch (IOException e) {
      e.printStackTrace();
    }
  }
}
```

DataSaveService.java

```
package uvic499.software.ecg;

import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.util.concurrent.LinkedBlockingQueue;
import java.util.concurrent.TimeUnit;

import android.app.Service;
import android.content.Context;
import android.content.Intent;
import android.graphics.Path;
import android.os.IBinder;

public class DataSaveService extends Service {

  // TODO: make package visibility so that ViewDataActivity can see it
  static DataThread writeThread;
  public LinkedBlockingQueue<Double> queue = BluetoothConnService.bluetoothQueueForSaving;

  @Override
  public int onStartCommand(Intent intent, int flags, int startId) {


    /*
     * thread:
     *   set up file (possibly based on user)
     *   save data in queue to file
     *   work until queue is empty.
     *
     * (extra): have a way of viewing the file
     */

    String filepath = getActiveSavePath();

    writeThread = new DataThread(filepath);
    writeThread.start();

    // Keep running, so sticky
    return START_STICKY;
  }
  /*TODO: encapsulate this cross-app
   * Gets the appropriate file to write to.  If multiple users are supported, should have a new file for
each
   * and some way to know who's file to access.
   *
   * For now, just use/overwrite the same file each time the service is started.
   */
  String getActiveSavePath() {
    Context appContext = getApplicationContext();
    return appContext.getString(R.string.default_save_file);

  }

  @Override
  public IBinder onBind(Intent arg0) {
    // TODO Auto-generated method stub
    return null;
  }

  @Override
  public void onDestroy() {
    writeThread.cancel();
    super.onDestroy();
  }

  class DataThread extends Thread {
```

```
    private final FileOutputStream outStream;
    private boolean continueWriting = true;

    public DataThread(String filepath) {
      FileOutputStream tmp = null;
      try{
        tmp = openFileOutput(filepath, Context.MODE_PRIVATE);
      } catch (FileNotFoundException e){
        System.out.println(e);
      }
      outStream = tmp;

      // Make sure BluetoothConnService has run first and initialized the queue before trying to run
      try {
        while (queue == null) {
          Thread.sleep(2000);
        }
      } catch(InterruptedException e) {}
    }

    @Override
    public void run() {
      System.out.println("---MY ACTIVE LISTS:");
      for (String s : getApplicationContext().fileList())
        System.out.println("\t" + s);
      while (continueWriting) {
        try {
          // Wait on the queue for up to 5-seconds, since it doesn't matter if we are slow in data
writing
          Double d = queue.poll(5, TimeUnit.SECONDS);
          if (d == null) continue;
          outStream.write(d.toString().getBytes());
          // write a new-line character too!
          outStream.write("\n".getBytes());
        } catch (InterruptedException e){
          break;
        } catch (IOException e) {
          System.out.println(e+"\nError writing to file!");
          break;
        }
      }

      try {
        outStream.close();
      } catch (IOException e) {
        System.out.println(e+"\nError closing file");
      }
    }

    public void cancel() {
      continueWriting = false;
    }
  }

}
```

DataSaving.java

```
package uvic499.software.ecg;

import java.util.concurrent.LinkedBlockingQueue;


// TODO: Make this a static class to hold data/file saving stuff
```

```
public class DataSaving {

  public LinkedBlockingQueue<Double> queue = BluetoothConnService.bluetoothQueueForSaving;

  /*
   * Saves history queue data to file - empties the queue in the process
   */
  static void startSave(String savePath) {
    /*
    //TODO: Saving data to file and Reading from file should be synchronized
    synchronized(MainActivity.fileAccess) {
      FileOutputStream outStream = null;
      try{
        outStream = openFileOutput(savePath, Context.MODE_PRIVATE);
      } catch (FileNotFoundException e){
        System.out.println("---saveFile not found: "+e);
        return;
      }

      while (!queue.isEmpty()) {
        try {
          Double d = queue.poll();
          if (d == null) return; // We shouldn't get here
          outStream.write(d.toString().getBytes());
          // write a new-line character too!
          outStream.write("\n".getBytes());
        } catch (IOException e) {
          System.out.println(e+"\nError writing to file!");
          break;
        }
      }

      try {
        outStream.close();
      } catch (IOException e) {
        System.out.println(e+"\nError closing file");
      }
    }
    */
  }

}
```

ECGChartActivity.java

```
package uvic499.software.ecg;

import java.util.concurrent.LinkedBlockingQueue;
import java.util.concurrent.TimeUnit;

import org.achartengine.ChartFactory;
import org.achartengine.GraphicalView;
import org.achartengine.model.XYMultipleSeriesDataset;
import org.achartengine.model.XYSeries;
import org.achartengine.renderer.XYMultipleSeriesRenderer;
import org.achartengine.renderer.XYSeriesRenderer;

import android.app.Activity;
import android.graphics.Color;
import android.os.Bundle;
import android.os.Handler;
import android.os.Message;
import android.view.Menu;
```

```
import android.view.MenuInflater;

public class ECGChartActivity extends Activity {

   private XYSeries xySeries;
   private XYMultipleSeriesDataset dataset;
   private XYMultipleSeriesRenderer renderer;
   private XYSeriesRenderer rendererSeries;
   private GraphicalView view;
   private double samplingRate = 0.6;
   private int pointsToDisplay = 75;

   private int yMax = 600;
   private int yMin = 500;
   private int xScrollAhead = 35;
   private double currentX = 0;
   private final int chartDelay = 10; //  millisecond delay for count
   private ChartingThread chartingThread;

   // TODO: package visibility so that queue service can see when it is ready for data
   static boolean isActive = false;

   public LinkedBlockingQueue<Double> queue = BluetoothConnService.bluetoothQueueForUI;

   @Override
   public void onCreate(Bundle savedInstanceState) {
     super.onCreate(savedInstanceState);
     setContentView(R.layout.activity_chart);

     setChartLook();
     dataset = new XYMultipleSeriesDataset();
     xySeries = new XYSeries(renderer.getChartTitle());
     dataset.addSeries(xySeries);
     view = ChartFactory.getLineChartView(this, dataset, renderer);
     view.refreshDrawableState();
     currentX = 0; // reset the horizontal of the graphing

     setContentView(view);

     if (savedInstanceState != null) {
       currentX = savedInstanceState.getDouble("currentX");

     }

     // To deal with onCreate coming from orientation change, only create chartingThread first time
     if (chartingThread == null) {
       ChartHandler chartUIHandler = new ChartHandler();
       chartingThread = new ChartingThread(chartUIHandler);
       chartingThread.start();
     }
     isActive = true;
   }

   @Override
   protected void onSaveInstanceState(Bundle b) {
    super.onSaveInstanceState(b);
    b.putDouble("currentX", currentX);
    // now stop the charting thread
    chartingThread.cancel();
    try {
      chartingThread.join();
    } catch (InterruptedException e) {
      // Not sure if this kills thread
      chartingThread.interrupt();
    }
   }

   @Override
   protected void onDestroy() {
```

```
   super.onDestroy();
   if (this.isFinishing()) { // real stoppage
     isActive = false;
     chartingThread.cancel();
   } else { // orientation change
     // Keep threads alive on orientation change
   }

 }

@Override
public boolean onCreateOptionsMenu(Menu menu) {
  // Inflate the menu; this adds items to the action bar if it is present.
  getMenuInflater().inflate(R.menu.activity_main, menu);
  return true;
}

public void setChartLook() {
   renderer = new XYMultipleSeriesRenderer();
   renderer.setApplyBackgroundColor(true);
   renderer.setBackgroundColor(Color.BLACK);//argb(100, 50, 50, 50));
   renderer.setLabelsTextSize(35);
   renderer.setLegendTextSize(35);
   renderer.setAxesColor(Color.GRAY);
   renderer.setAxisTitleTextSize(35);
   renderer.setChartTitle("ECG Heartbeat");
   renderer.setChartTitleTextSize(35);
   renderer.setFitLegend(false);
   renderer.setGridColor(Color.BLACK);
   renderer.setPanEnabled(false, false); // TODO
   renderer.setPointSize(1);
   renderer.setXTitle("X");
   renderer.setYTitle("Y");
   renderer.setMargins(new int []{20, 20, 50, 20}); // TODO: i doubled
   renderer.setZoomButtonsVisible(false);
   renderer.setZoomEnabled(false);
   renderer.setBarSpacing(10);
   renderer.setShowGrid(false);
   renderer.setYAxisMax(yMax);
   renderer.setYAxisMin(yMin);
   //TODO: Don't show labels or legend
   renderer.setShowLabels(false);;
   renderer.setShowLegend(false);

   rendererSeries = new XYSeriesRenderer();
   rendererSeries.setColor(Color.GREEN);
   rendererSeries.setLineWidth(10f);
   renderer.addSeriesRenderer(rendererSeries);
}

//
class ChartHandler extends Handler {
  @Override
  public void handleMessage(Message msg) {
      double yVal = ((double)msg.arg1)/1000;
      xySeries.add(currentX, yVal);
       if (currentX - pointsToDisplay >= 0 ) {
         renderer.setXAxisMin(currentX - pointsToDisplay);
       }
       renderer.setXAxisMax(currentX + xScrollAhead);
       view.repaint();
    }
}

class ChartingThread extends Thread {
  private boolean continueCharting = true;
  private Handler handler;
  public ChartingThread(Handler handler) {
    this.handler = handler;
```

```
    }

    @Override
    public void run() {
        while(continueCharting) {
         Double yVal = null;
          try {
            Thread.sleep(chartDelay);
            yVal = queue.poll(2, TimeUnit.SECONDS);
          } catch (InterruptedException e) {
            e.printStackTrace();
            continue;
          }
            // skip if no data on queue
            if (yVal == null) {
              continue;
            }
            currentX = currentX+samplingRate;
         if (yVal > yMax) {
           yMax = yVal.intValue();
           renderer.setYAxisMax(yVal);
         } else if (yVal < yMin) {
           yMin = yVal.intValue();
           renderer.setYAxisMin(yVal);
         }

            // send Message to UI handler for charting.
            Message msg = Message.obtain();
            // Send as an integer. Handler converts it back to an integer
            msg.arg1 = (int)Math.round(yVal*1000);
            handler.sendMessage(msg);
          }
      }

    // Stops the thread
    public void cancel() {
      continueCharting = false;
    }
  }

}
```

MainActivity.java

```
package uvic499.software.ecg;

import java.io.FileOutputStream;
import java.lang.reflect.Field;
import java.util.Calendar;
import java.util.Queue;

import android.app.Activity;
import android.bluetooth.BluetoothAdapter;
import android.content.Context;
import android.content.Intent;
import android.os.Bundle;
import android.view.Menu;
import android.view.View;
import android.widget.Toast;

public class MainActivity extends Activity {

  protected final static int REQUEST_ENABLE_BLUETOOTH = 1;
  private BluetoothAdapter BT = BluetoothAdapter.getDefaultAdapter();
```

```java
private String deviceName;
private String deviceMAC;
private String documentSavePath = "";

private Intent btConnServiceIntent;

private Queue<Double> queue = BluetoothConnService.bluetoothQueueForSaving;

@Override
protected void onCreate(Bundle savedInstanceState) {

  super.onCreate(savedInstanceState);
  setContentView(R.layout.activity_main);
  toggleUI(false);

  btConnServiceIntent = new Intent(MainActivity.this, BluetoothConnService.class);
  // start services only if Bluetooth is on
  if (BT.isEnabled()) {
    startService(btConnServiceIntent);
  }

  checkBluetooth();

  if (savedInstanceState != null) {
    documentSavePath = savedInstanceState.getString("documentSavePath");
  }
}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
  // Inflate the menu; this adds items to the action bar if it is present.
  getMenuInflater().inflate(R.menu.activity_main, menu);
  return true;
}

// Stop the Bluetooth service when the app is closed
// NOTE: do nothing onStop, as it is called when Activity loses focus -
// in this case App is still running, so we want service to keep running too
@Override
public void onDestroy() {
  super.onDestroy();
  if (isFinishing()) {
    saveBluetoothDataToFile();
    // Definitely want to stop Bluetooth connections
    stopService(btConnServiceIntent);
  } else {
    // just an orientation change - do nothing
  }
}

@Override
protected void onSaveInstanceState(Bundle b) {
  super.onSaveInstanceState(b);
  b.putString("documentSavePath", documentSavePath);
}

// Turns Bluetooth on if it is off.  Starts BT service handler afterwards
private void checkBluetooth() {
  if (!BT.isEnabled()) {
    // Bluetooth not enabled: display a message and prompt user to turn it on
    Intent enableBT = new Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
    startActivityForResult(enableBT, REQUEST_ENABLE_BLUETOOTH);
  } else {
    toggleUI(true);
  }
}

protected void onActivityResult(int requestCode, int resultCode, Intent data) {
  if (requestCode == REQUEST_ENABLE_BLUETOOTH) {
```

```
    if (resultCode == Activity.RESULT_OK) {
      deviceMAC = BT.getAddress();
      deviceName = BT.getName();
      String toastText = "Bluetooth enabled for device\n" + deviceName + " : " + deviceMAC;
      // turn UI on and show success msg
      toggleUI(true);
      Toast.makeText(this, toastText, Toast.LENGTH_LONG).show();

      // Now start Bluetooth
      startService(btConnServiceIntent);

    } else if (resultCode == Activity.RESULT_CANCELED) {
      System.out.println("RESULT_CANCELLED!!!");
      toggleUI(false);
    }
  }
}

// Chart Button onClick
public void startChart(View v) {
  Intent intent = new Intent(this, ECGChartActivity.class);
  startActivity(intent);
}

// View Data Button onClick
public void viewSavedData(View v) {
  Intent intent = new Intent(this, ViewSavedDataActivity.class);
  startActivity(intent);
}

private void saveBluetoothDataToFile() {

  Calendar calendar = Calendar.getInstance();
  String date = calendar.getTime().toString();

  System.out.println("----savepath ="+date);
  FileOutputStream outputStream;

  try {
    outputStream = openFileOutput(date, Context.MODE_PRIVATE);

    for (Double d : queue) {
      if (d == null) continue;

      for (char c : String.valueOf(d).toCharArray()) {
        outputStream.write((int)c);
      }
      // Print new line after each
      outputStream.write('\n');
      System.out.println("---wrote double "+d);
    }
    outputStream.close();
  } catch (Exception e) {
    e.printStackTrace();
  }
}

private void toggleUI(boolean enabled) {
  System.out.println("Toggling UI to " + enabled);
  int visibility = enabled ? View.VISIBLE : View.GONE;
  for (Field f : R.id.class.getFields()) {
    try {
      int id = f.getInt(null);

      // If BT connection msg, do opposite action, and always hide progress bar
      View v = this.findViewById(id);

      if (id == R.id.Bluetooth_connection) {
        v.setEnabled(!enabled);
```

```
          int tempVis = enabled ? View.GONE : View.VISIBLE;
          v.setVisibility(tempVis);
        } else {
          v.setEnabled(enabled);
          v.setVisibility(visibility);
        }
      } catch (Exception e) {
        // TODO: just keep going through all fields... why not?
        System.out.println(e);
      }
    }
  }

}
```

ViewSavedDataActivity.java

```
package uvic499.software.ecg;

import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.util.ArrayList;
import java.util.concurrent.LinkedBlockingQueue;

import org.achartengine.ChartFactory;
import org.achartengine.GraphicalView;
import org.achartengine.model.XYMultipleSeriesDataset;
import org.achartengine.model.XYSeries;
import org.achartengine.renderer.XYMultipleSeriesRenderer;
import org.achartengine.renderer.XYSeriesRenderer;

import android.app.Activity;
import android.graphics.Color;
import android.os.Bundle;

public class ViewSavedDataActivity extends Activity {

   private XYMultipleSeriesRenderer renderer;
   private XYSeriesRenderer rendererSeries;
   private XYSeries xySeries;
   private XYMultipleSeriesDataset dataset;
   private GraphicalView view;

   // TODO: The sampling rate here needs to be accurate (it's not now)
   private double samplingRate = 1;
   private int xLookahead = 20;
   private int yMax = 600;
   private int yMin = 500;

   private ArrayList<Double> savedData = new ArrayList<Double>();
   public LinkedBlockingQueue<Double> queue = BluetoothConnService.bluetoothQueueForSaving;

  /* TODO: encapsulate this cross-app
   * Gets the appropriate file to write to.  If multiple users are supported, should have a new file for
each
   * and some way to know who's file to access.
   *
   * For now, just use/overwrite the same file each time the service is started.
   */

  @Override
  public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_chart);
```

```
  // TODO: for now, saved data should just fill with in-memory history queue
  for (Object d : queue.toArray()) {
    savedData.add((Double)d);
  }

  displaySavedData();
}


// Puts all file data into main memory array - TODO: NOT USED
public void readSavedData(String filepath) {

    FileInputStream fin = null;
    try {
      fin = openFileInput(filepath);
    } catch(FileNotFoundException e) {
      System.out.println("File " + filepath + " does not exist.");
      return;
    }
    char c;
    char doubleBuilder[] = new char[7];
    int i = 0;
    boolean continueReading = true;
    while (continueReading) {
      try{
        if (fin.available() > 0) {
          c = (char)fin.read();
          if (c == '\n' || i > 6) {
            // end of the double, put it in array
            Double d = Double.valueOf(String.copyValueOf(doubleBuilder));
            //System.out.println("--- queueing:" + String.copyValueOf(doubleBuilder));
            savedData.add(d);
            i = 0;
          } else {
            doubleBuilder[i++] = c;
          }
        } else {
          // Finished reading file
          continueReading = false;
        }
      } catch (IOException e) {
        System.out.println(e);
        continueReading = false;
      }
    }

    try {
      fin.close();
    } catch(IOException e) {
    }
  }

public void displaySavedData() {

  initChartStuff();
  dataset = new XYMultipleSeriesDataset();
  xySeries = new XYSeries(renderer.getChartTitle());

  // fill series
  double currentX = 0;
  for (Double d : savedData) {
    xySeries.add(currentX, d);
    currentX += samplingRate;
    if (d > yMax) {
      yMax = d.intValue();
    }
    if ( d < yMin) {
      yMin = d.intValue();
```

```
      }
    }

    dataset.addSeries(xySeries);

    // TODO: Setup the X axis max and mins once we have a filled series
    int xCurr = xySeries.getItemCount();
    int xMin = (xCurr-400 >= 0) ? xCurr-400 : 0;
    renderer.setXAxisMax(xCurr+xLookahead);
    renderer.setXAxisMin(xMin);
    renderer.setYAxisMax(yMax);
    renderer.setYAxisMin(yMin);


    view = ChartFactory.getLineChartView(getApplicationContext(), dataset, renderer);
    view.refreshDrawableState();
    setContentView(view);

  }

  // TODO: This stuff taken from ECGChartActivity
  private void initChartStuff() {
    renderer = new XYMultipleSeriesRenderer();
    renderer.setApplyBackgroundColor(true);
    renderer.setBackgroundColor(Color.BLACK);//argb(100, 50, 50, 50));
    renderer.setLabelsTextSize(35);
    renderer.setLegendTextSize(35);
    renderer.setAxesColor(Color.WHITE);
    renderer.setAxisTitleTextSize(35);
    renderer.setChartTitle("ECG Heartbeat");
    renderer.setChartTitleTextSize(35);
    renderer.setFitLegend(false);
    renderer.setGridColor(Color.BLACK);
    renderer.setPanEnabled(true, true); // TODO
    renderer.setPointSize(1);
    renderer.setXTitle("X");
    renderer.setYTitle("Y");
    renderer.setMargins(new int []{5, 50, 50, 5}); // TODO: i doubled
    renderer.setZoomButtonsVisible(false);
    renderer.setZoomEnabled(true); // TODO: try true
    renderer.setBarSpacing(10);
    renderer.setShowGrid(false);

    // TODO: Reset the MAX AND MIN VALUES!!
    renderer.setYAxisMax(yMax);//2.4);
    renderer.setYAxisMin(yMin);//0.4);

    rendererSeries = new XYSeriesRenderer();
    rendererSeries.setColor(Color.GREEN);
    rendererSeries.setLineWidth(5f);
    renderer.addSeriesRenderer(rendererSeries);
  }

}
```

Appendix B: Firmware Source Code

```
import Bluetooth as bt
import time
import sys
import os
import spidev
import threading
import Queue

spi = spidev.SpiDev()
spi.open(0, 0)

valueQueue = Queue.Queue()

def readADC():
	# read SPI data from MCP3008 chip, channel 0
	r = spi.xfer2([1, 8 << 4, 0])
	adcout = ((r[1] & 3) << 8) + r[2]
	return adcout

def startReading():
	while keepReading:
		for i in range(0,25):
			readADC()
		valueQueue.put(readADC())

server_UUID = "00001101-0000-1000-8000-00805F9B34FA"

result = bt.find_service(name="ECG Monitor", uuid=server_UUID,address=server_MAC)

if len(result) < 1:
	print("No service found!")
	sys.exit(0)

result = result[0]
print result

host = result.get("host")
port = result.get("port")

sock = bt.BluetoothSocket(bt.RFCOMM)

keepReading = True

try:
	sock.connect((host, port))
except Exception as e:
	print e
```

Appendix B: Firmware Source Code

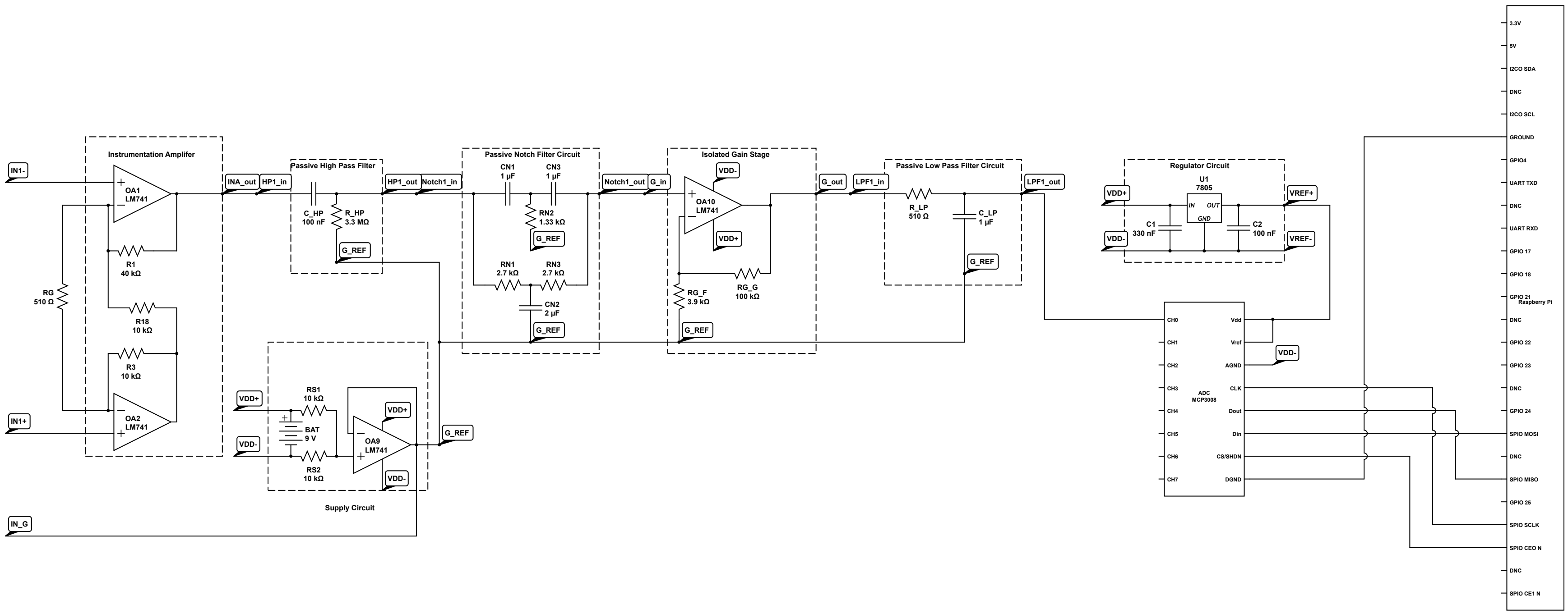
```
else:
	t = threading.Thread(target=startReading)
	t.daemon = True
	t.start()

	keepSending = True
	while keepSending is True:

		value = valueQueue.get(block=True)
		if value < 0 or value > 1024:
			# value is corrupt, get a new one
			continue
		try:
			print value
			sock.send(str(value) + '\n')
		except bt.BluetoothError as BTError:
			print "Error sending data: " + str(BTError)

			keepSending = False
finally:
	sock.close()
	keepReading = False
	t.join()
```

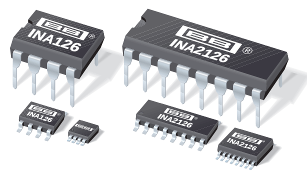
Appendix C: Hardware Circuit



Instrumentation Amplifer
IN1-
IN1+
OA1
LM741
OA2
LM741
R1
40 kΩ
R18
10 kΩ
R3
10 kΩ
RG
510 Ω
INA_out
HP1_in
Passive High Pass Filter
C_HP
100 nF
R_HP
3.3 MΩ
G_REF
HP1_out
Notch1_in
Passive Notch Filter Circuit
CN1
1 μF
CN3
1 μF
RN2
1.33 kΩ
RN1
2.7 kΩ
RN3
2.7 kΩ
CN2
2 μF
Notch1_out
G_in
Isolated Gain Stage
VDD-
VDD+
OA10
LM741
RG_F
3.9 kΩ
RG_G
100 kΩ
G_out
LPF1_in
Passive Low Pass Filter Circuit
R_LP
510 Ω
C_LP
1 μF
LPF1_out
Regulator Circuit
U1
7805
IN
OUT
GND
C1
330 nF
C2
100 nF
VREF+
VREF-
Supply Circuit
RS1
10 kΩ
BAT
9 V
RS2
10 kΩ
OA9
LM741
IN_G
ADC
MCP3008
CH0
CH1
CH2
CH3
CH4
CH5
CH6
CH7
Vdd
Vref
AGND
CLK
Dout
Din
CS/SHDN
DGND
Raspberry Pi
3.3V
5V
I2CO SDA
DNC
I2CO SCL
GROUND
GPIO4
UART TXD
UART RXD
GPIO 17
GPIO 18
GPIO 21
GPIO 22
GPIO 23
GPIO 24
SPIO MOSI
SPIO MISO
GPIO 25
SPIO SCLK
SPIO CEO N
SPIO CE1 N

Appendix D: Datasheets for Hardware Components

BURR - BROWN ®



INA126
INA2126

*Micro*POWER INSTRUMENTATION AMPLIFIER
Single and Dual Versions

FEATURES

- LOW QUIESCENT CURRENT: 175µA/chan.
- WIDE SUPPLY RANGE: ±1.35V to ±18V
- LOW OFFSET VOLTAGE: 250µV max
- LOW OFFSET DRIFT: 3µV/°C max
- LOW NOISE: 35nV/$\sqrt{\text{Hz}}$
- LOW INPUT BIAS CURRENT: 25nA max
- 8-PIN DIP, SO-8, MSOP-8 SURFACE- MOUNT DUAL: 16-Pin DIP, SO-16, SSOP-16

APPLICATIONS

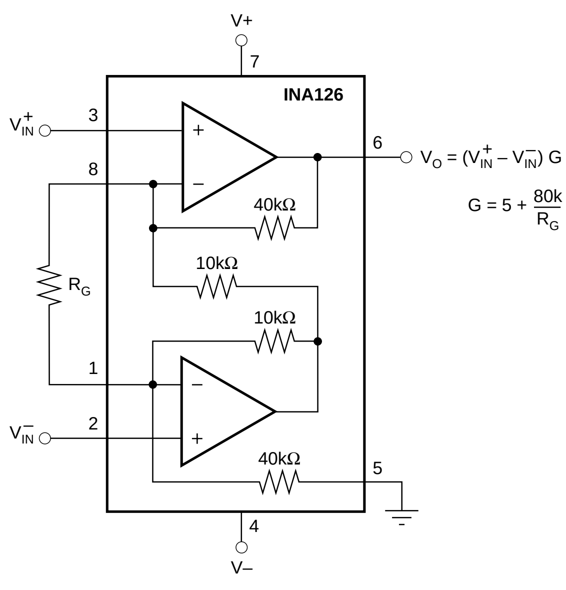
- INDUSTRIAL SENSOR AMPLIFIER: Bridge, RTD, Thermocouple
- PHYSIOLOGICAL AMPLIFIER: ECG, EEG, EMG
- MULTI-CHANNEL DATA ACQUISITION
- PORTABLE, BATTERY OPERATED SYSTEMS

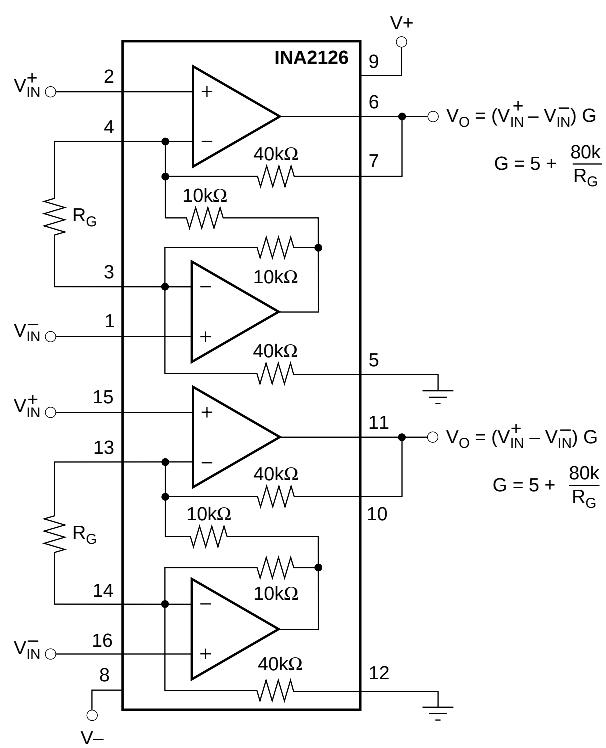
DESCRIPTION

The INA126 and INA2126 are precision instrumentation amplifiers for accurate, low noise differential signal acquisition. Their two-op-amp design provides excellent performance with very low quiescent current (175µA/chan.). This, combined with wide operating voltage range of ±1.35V to ±18V, makes them ideal for portable instrumentation and data acquisition systems.

Gain can be set from 5V/V to 10000V/V with a single external resistor. Laser trimmed input circuitry provides low offset voltage (250µV max), low offset voltage drift (3µV/°C max) and excellent common-mode rejection.

Single version package options include 8-pin plastic DIP, SO-8 surface mount, and fine-pitch MSOP-8 surface-mount. Dual version is available in the space-saving SSOP-16 fine-pitch surface mount, SO-16, and 16-pin DIP. All are specified for the –40°C to +85°C industrial temperature range.





International Airport Industrial Park • Mailing Address: PO Box 11400, Tucson, AZ 85734 • Street Address: 6730 S. Tucson Blvd., Tucson, AZ 85706 • Tel: (520) 746-1111 • Twx: 910-952-1111
Internet: http://www.burr-brown.com/ • FAXLine: (800) 548-6133 (US/Canada Only) • Cable: BBRCORP • Telex: 066-6491 • FAX: (520) 889-1510 • Immediate Product Info: (800) 548-6132

 PDS-1365C Printed in U.S.A. September, 1997

SPECIFICATIONS

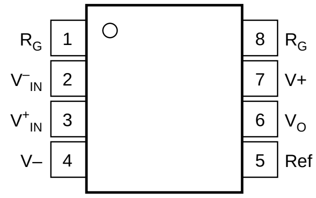
At T_A = +25°C, V_S = ±15V, R_L = 25kΩ, unless otherwise noted.

PARAMETER	CONDITIONS	INA126P, U, E INA2126P, U, E MIN	TYP	MAX	INA126PA, UA, EA INA2126PA, UA, EA MIN	TYP	MAX	UNITS
INPUT								
Offset Voltage, RTI			±100	±250		±150	±500	µV
vs Temperature			±0.5	±3		*	±5	µV/°C
vs Power Supply (PSRR)	V_S = ±1.35V to ±18V		5	15		*	50	µV/V
Input Impedance			10^9 || 4			*		Ω || pF
Safe Input Voltage	R_S = 0	(V–)–0.5		(V+)+0.5	*		*	V
	R_S = 1kΩ	(V–)–10		(V+)+10	*		*	V
Common-Mode Voltage Range	V_O = 0V	±11.25	±11.5		*	*		V
Channel Separation (dual)	G = 5, dc		130					dB
Common-Mode Rejection	R_S = 0, V_{CM} = ±11.25V	83	94		74	90		dB
INA2126U (dual SO-16)		80	94					dB
INPUT BIAS CURRENT			–10	–25		*	–50	nA
vs Temperature			±30			*		pA/°C
Offset Current			±0.5	±2		*	±5	nA
vs Temperature			±10			*		pA/°C
GAIN			G = 5 to 10k			*		V/V
Gain Equation			G = 5 + 80kΩ/R_G			*		V/V
Gain Error	V_O = ±14V, G = 5		±0.02	±0.1		*	±0.18	%
vs Temperature	G = 5		±2	±10		*	*	ppm/°C
Gain Error	V_O = ±12V, G = 100		±0.2	±0.5		*	±1	%
vs Temperature	G = 100		±25	±100		*	*	ppm/°C
Nonlinearity	G = 100, V_O = ±14V		±0.002	±0.012		*	*	%
NOISE								
Voltage Noise, f = 1kHz			35			*		nV/√Hz
f = 100Hz			35			*		nV/√Hz
f = 10Hz			45			*		nV/√Hz
f_B = 0.1Hz to 10Hz			0.7			*		µVp-p
Current Noise, f = 1kHz			60			*		fA/√Hz
f_B = 0.1Hz to 10Hz			2			*		pAp-p
OUTPUT								
Voltage, Positive	R_L = 25kΩ	(V+)–0.9	(V+)–0.75		*	*		V
Negative	R_L = 25kΩ	(V–)+0.95	(V–)+0.8		*	*		V
Short-Circuit Current	Short-Circuit to Ground		+10/–5			*		mA
Capacitive Load Drive			1000			*		pF
FREQUENCY RESPONSE								
Bandwidth, –3dB	G = 5		200			*		kHz
	G = 100		9			*		kHz
	G = 500		1.8			*		kHz
Slew Rate	V_O = ±10V, G = 5		0.4			*		V/µs
Settling Time, 0.01%	10V Step, G = 5		30			*		µs
	10V Step, G = 100		160			*		µs
	10V Step, G = 500		1500			*		µs
Overload Recovery	50% Input Overload		4			*		µs
POWER SUPPLY								
Voltage Range		±1.35	±15	±18	*	*	*	V
Current (per channel)	I_O = 0		±175	±200		*	*	µA
TEMPERATURE RANGE								
Specification Range		–40		+85	*		*	°C
Operation Range		–55		+125	*		*	°C
Storage Range		–55		+125	*		*	°C
Thermal Resistance, θ_{JA}								
8-Pin DIP			100			*		°C/W
SO-8 Surface-Mount			150			*		°C/W
MSOP-8 Surface-Mount			200			*		°C/W
16-Pin DIP (dual)			80			*		°C/W
SO-16 (dual)			100			*		°C/W
SSOP-16 (dual)			100			*		°C/W

* Specification same as INA126P, INA126U, INA126E; INA2126P, INA2126U, INA2126E.

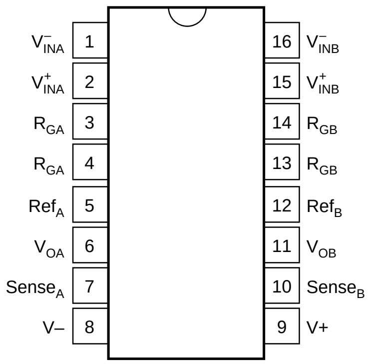
PIN CONFIGURATION (Single)

Top View — 8-Pin DIP, SO-8, MSOP-8



PIN CONFIGURATION (Dual)

Top View — 16-Pin DIP, SO-16, SSOP-16



ABSOLUTE MAXIMUM RATINGS[1]

Power Supply Voltage, V+ to V–	36V
Input Signal Voltage[2]	(V–)–0.7 to (V+)+0.7V
Input Signal Current[2]	10mA
Output Short Circuit	Continuous
Operating Temperature	–55°C to +125°C
Storage Temperature	–55°C to +125°C
Lead Temperature (soldering, 10s)	+300°C

NOTES: (1) Stresses above these ratings may cause permanent damage. (2) Input signal voltage is limited by internal diodes connected to power supplies. See text.

ELECTROSTATIC DISCHARGE SENSITIVITY

This integrated circuit can be damaged by ESD. Burr-Brown recommends that all integrated circuits be handled with appropriate precautions. Failure to observe proper handling and installation procedures can cause damage.

ESD damage can range from subtle performance degradation to complete device failure. Precision integrated circuits may be more susceptible to damage because very small parametric changes could cause the device not to meet its published specifications.

PACKAGE INFORMATION

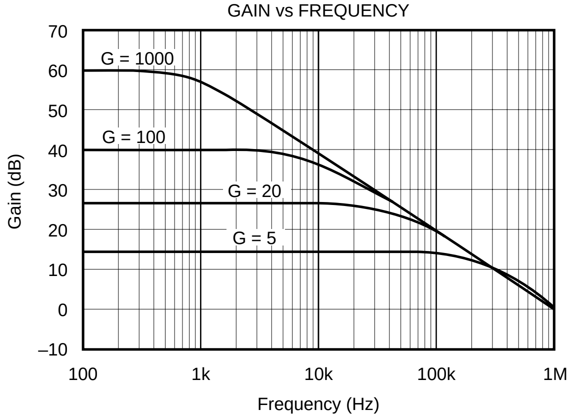
PRODUCT	PACKAGE	PACKAGE DRAWING NUMBER[1]	PACKAGE MARKING	ORDERING NUMBER	TRANSPORT MEDIA
Single					
INA126PA INA126P	8-Pin DIP 8-Pin DIP	006 006	INA126PA INA126P	INA126PA INA126P	Rails Rails
INA126UA INA126U	SO-8 SO-8	182 182	INA126UA INA126U	INA126UA INA126U	Rails or Reel Rails or Reel
INA126EA[2] " INA126E[2] "	MSOP-8 " MSOP-8 "	337 " 337 "	A26[3] " A26[3] "	INA126EA-250 INA126EA-2500 INA126E-250 INA126E-2500	Reel Only " Reel Only "
Dual					
INA2126PA INA2126P	16-Pin DIP 16-Pin DIP	180 180	INA2126PA INA2126P	INA2126PA INA2126P	Rails Rails
INA2126UA INA2126U	SO-16 SO-16	265 265	INA2126UA INA2126U	INA2126UA INA2126U	Rails Rails
INA2126EA[2] " INA2126E[2] "	SSOP-16 " SSOP-16 "	322 " 322 "	INA2126EA " INA2126E "	INA2126EA-250 INA2126EA-2500 INA2126E-250 INA2126E-2500	Reel Only " Reel Only "

NOTES: (1) For detailed drawing and dimension table, see end of data sheet, or Appendix C of Burr-Brown IC Data Book. (2) MSOP-8 and SSOP-16 packages are available only on 250 or 2500 piece reels. (3) Grade designation is marked on reel.

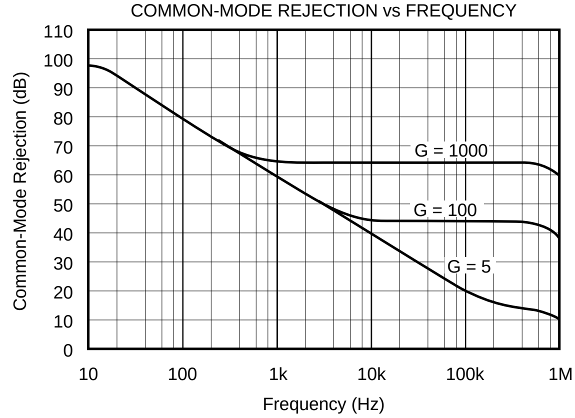
TYPICAL PERFORMANCE CURVES

At T_A = +25°C and V_S = ±15V, unless otherwise noted.

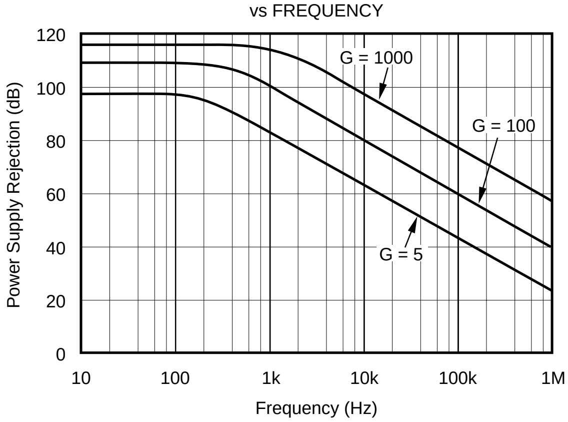
GAIN vs FREQUENCY



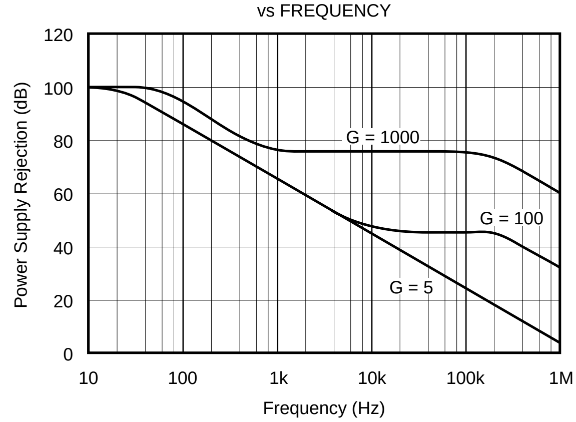
COMMON-MODE REJECTION vs FREQUENCY



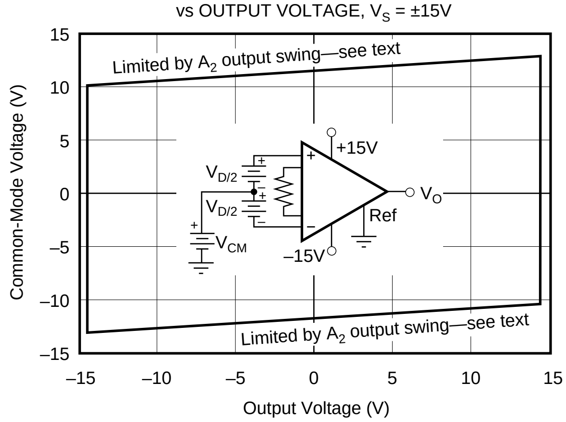
POSITIVE POWER SUPPLY REJECTION
vs FREQUENCY



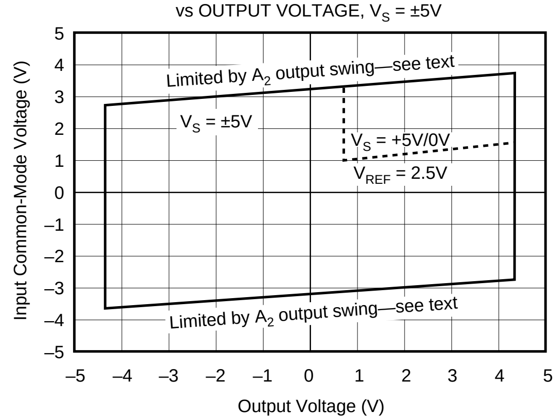
NEGATIVE POWER SUPPLY REJECTION
vs FREQUENCY



INPUT COMMON-MODE RANGE
vs OUTPUT VOLTAGE, V_S = ±15V



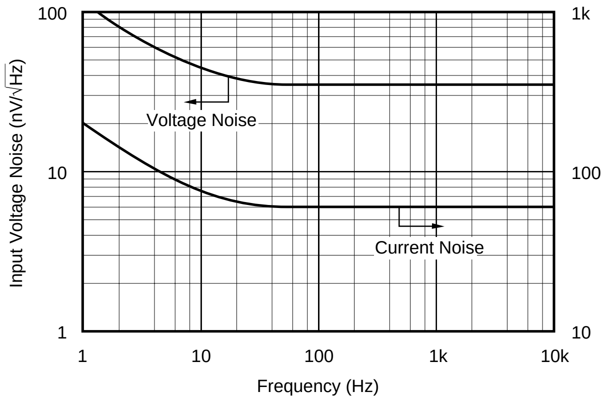
INPUT COMMON-MODE VOLTAGE RANGE
vs OUTPUT VOLTAGE, V_S = ±5V



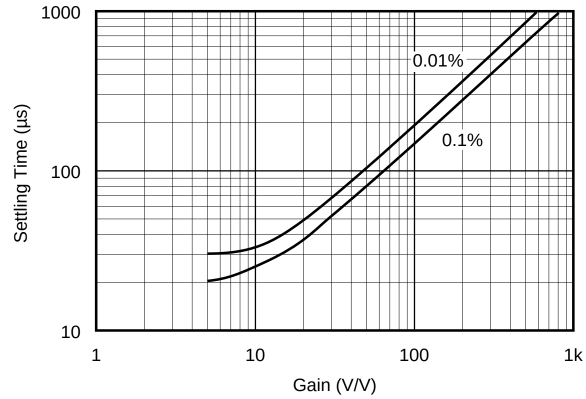
TYPICAL PERFORMANCE CURVES (CONT)

At $T_A = +25°C$ and $V_S = ±15V$, unless otherwise noted.

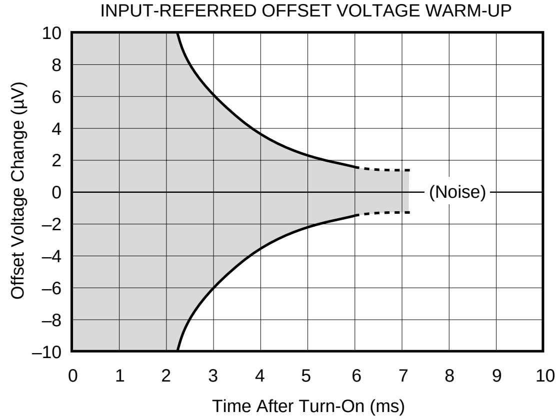
INPUT-REFERRED NOISE vs FREQUENCY



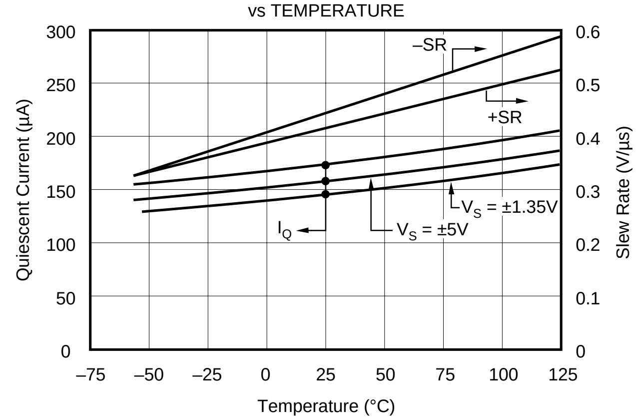
SETTLING TIME vs GAIN



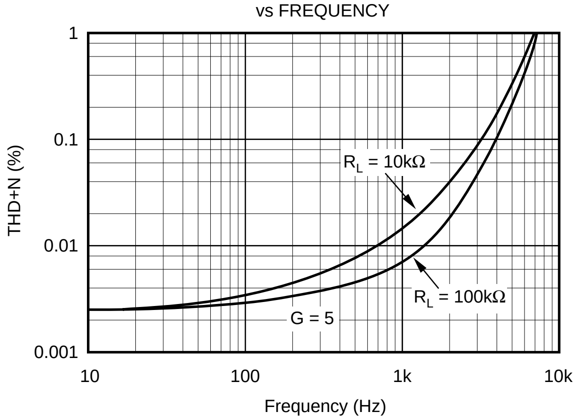
INPUT-REFERRED OFFSET VOLTAGE WARM-UP



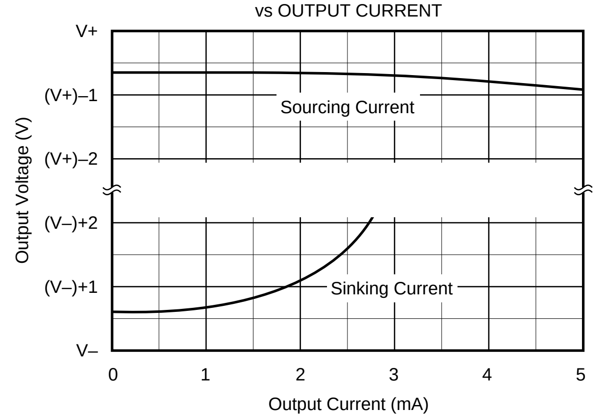
QUIESCENT CURRENT AND SLEW RATE
vs TEMPERATURE



TOTAL HARMONIC DISTORTION+NOISE
vs FREQUENCY



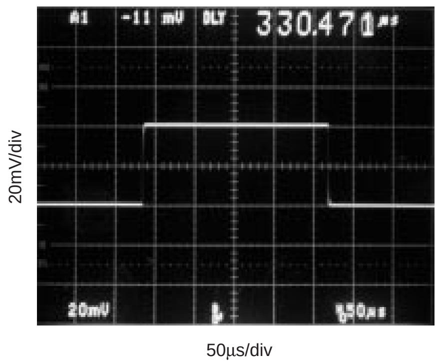
OUTPUT VOLTAGE SWING
vs OUTPUT CURRENT



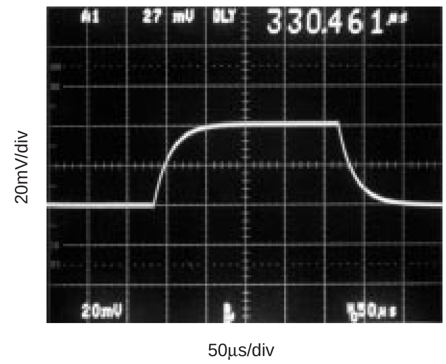
TYPICAL PERFORMANCE CURVES (CONT)

At $T_A = +25°C$ and $V_S = ±15V$, unless otherwise noted.

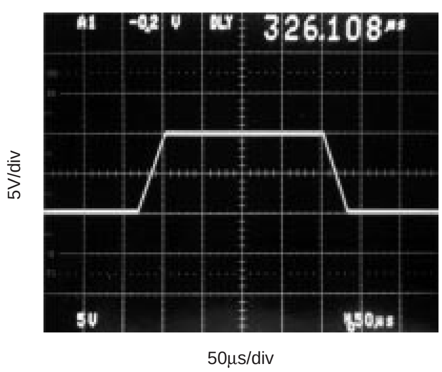
SMALL-SIGNAL RESPONSE, G = 5



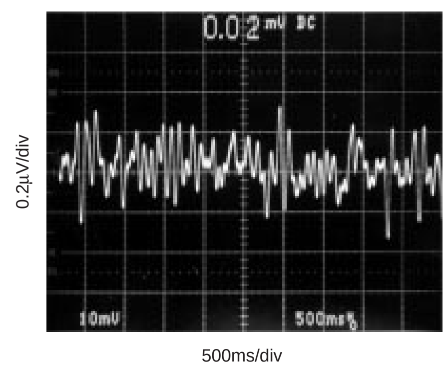
SMALL-SIGNAL RESPONSE, G = 100



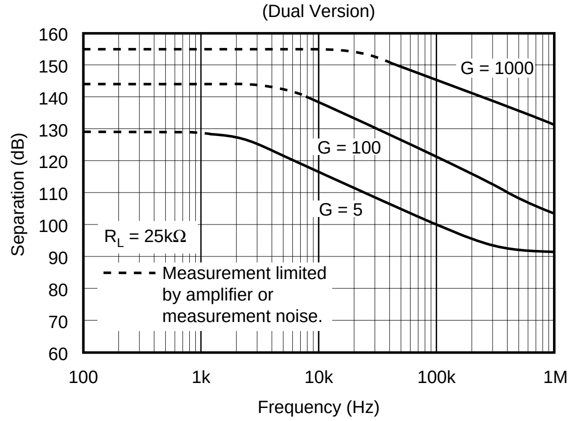
LARGE-SIGNAL RESPONSE, G = 5



VOLTAGE NOISE, 0.1Hz to 10Hz



CHANNEL SEPARATION vs FREQUENCY, RTI
(Dual Version)



APPLICATION INFORMATION

Figure 1 shows the basic connections required for operation of the INA126. Applications with noisy or high impedance power supplies may require decoupling capacitors close to the device pins as shown.

The output is referred to the output reference (Ref) terminal which is normally grounded. This must be a low-impedance connection to ensure good common-mode rejection. A resistance of 8Ω in series with the Ref pin will cause a typical device to degrade to approximately 80dB CMR.

Dual versions (INA2126) have feedback sense connections, $Sense_A$ and $Sense_B$. These must be connected to their respective output terminals for proper operation. The sense connection can be used to sense the output voltage directly at the load for best accuracy.

SETTING THE GAIN

Gain is set by connecting an external resistor, R_G, as shown:

$$G = 5 + \frac{80k\Omega}{R_G} \quad (1)$$

Commonly used gains and R_G resistor values are shown in Figure 1.

The 80kΩ term in equation 1 comes from the internal metal film resistors which are laser trimmed to accurate absolute values. The accuracy and temperature coefficient of these resistors are included in the gain accuracy and drift specifications.

The stability and temperature drift of the external gain setting resistor, R_G, also affects gain. R_G's contribution to gain accuracy and drift can be directly inferred from the gain equation (1). Low resistor values required for high gain can make wiring resistance important. Sockets add to the wiring resistance, which will contribute additional gain error in gains of approximately 100 or greater.

OFFSET TRIMMING

The INA126 and INA2126 are laser trimmed for low offset voltage and offset voltage drift. Most applications require no external offset adjustment. Figure 2 shows an optional circuit for trimming the output offset voltage. The voltage applied to the Ref terminal is added to the output signal. An op amp buffer is used to provide low impedance at the Ref terminal to preserve good common-mode rejection.

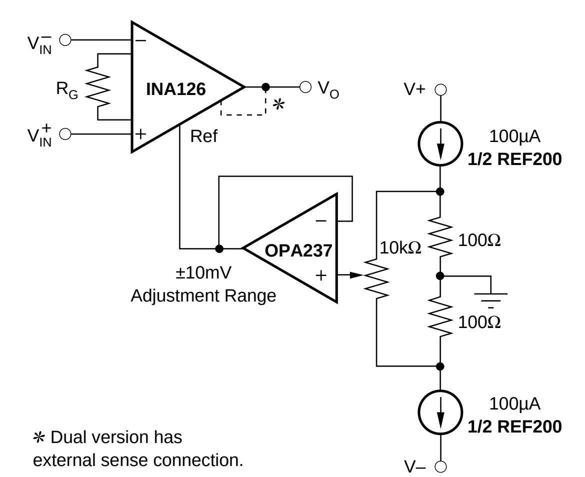
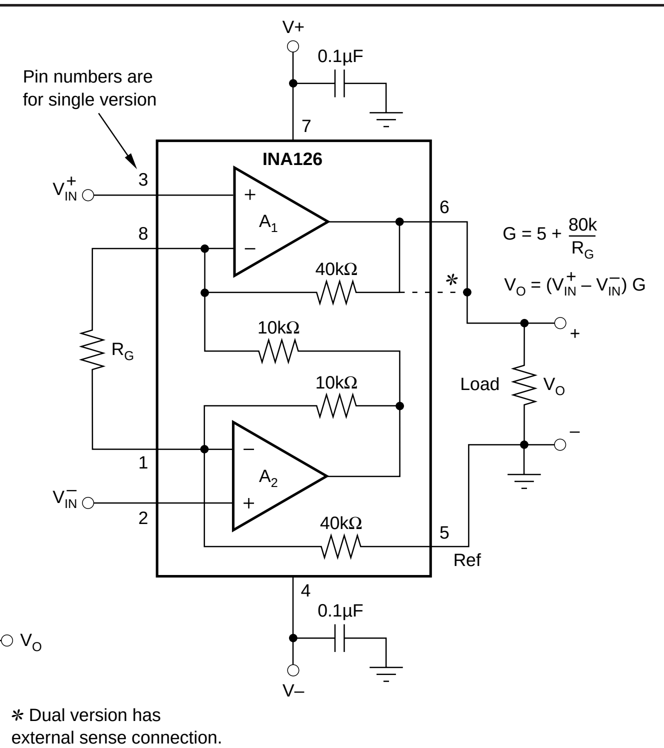


FIGURE 2. Optional Trimming of Output Offset Voltage.

DESIRED GAIN (V/V)	R_G (Ω)	NEAREST 1% R_G VALUE
5	NC	NC
10	16k	15.8k
20	5333	5360
50	1779	1780
100	842	845
200	410	412
500	162	162
1000	80.4	80.6
2000	40.1	40.2
5000	16.0	15.8
10000	8.0	7.87

NC: No Connection.



Also drawn in simplified form:

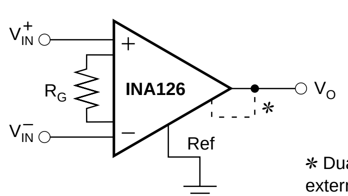


FIGURE 1. Basic Connections.

INPUT BIAS CURRENT RETURN

The input impedance of the INA126/2126 is extremely high—approximately $10^9\Omega$. However, a path must be provided for the input bias current of both inputs. This input bias current is typically –10nA (current flows out of the input terminals). High input impedance means that this input bias current changes very little with varying input voltage.

Input circuitry must provide a path for this input bias current for proper operation. Figure 3 shows various provisions for an input bias current path. Without a bias current path, the inputs will float to a potential which exceeds the common-mode range and the input amplifiers will saturate.

If the differential source resistance is low, the bias current return path can be connected to one input (see the thermocouple example in Figure 3). With higher source impedance, using two equal resistors provides a balanced input with advantages of lower input offset voltage due to bias current and better high-frequency common-mode rejection.

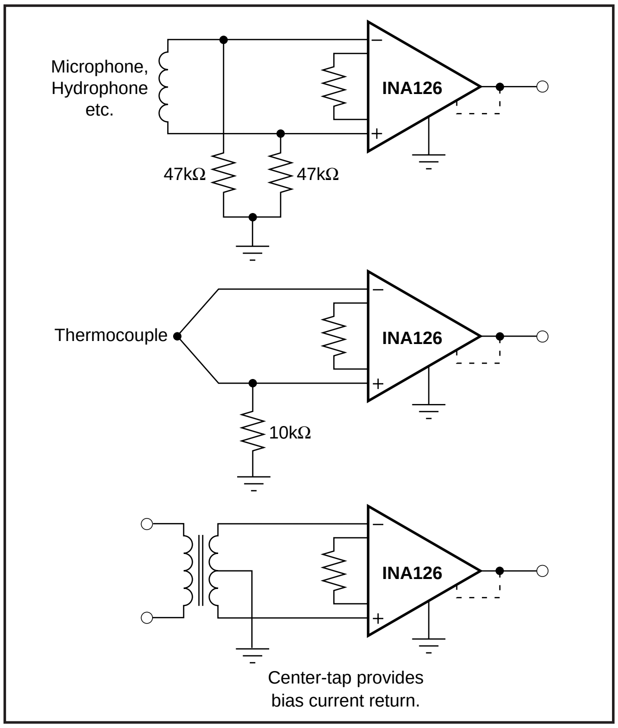


FIGURE 3. Providing an Input Common-Mode Current Path.

INPUT COMMON-MODE RANGE

The input common-mode range of the INA126/2126 is shown in typical performance curves. The common-mode range is limited on the negative side by the output voltage swing of A_2, an internal circuit node that cannot be measured on an external pin. The output voltage of A_2 can be expressed as:

$$V_{O2} = 1.25\ V_{IN}^{-} - (V_{IN}^{+} - V_{IN}^{-})\ (10k\Omega/R_G) \qquad (2)$$

(Voltages referred to Ref terminal, pin 5)

The internal op amp A_2 is identical to A_1 and its output swing is limited to typically 0.7V from the supply rails. When the input common-mode range is exceeded (A_2's output is saturated), A_1 can still be in linear operation and respond to changes in the non-inverting input voltage. The output voltage, however, will be invalid.

LOW VOLTAGE OPERATION

The INA126/2126 can be operated on power supplies as low as ±1.35V. Performance remains excellent with power supplies ranging from ±1.35V to ±18V. Most parameters vary only slightly throughout this supply voltage range—see typical performance curves. Operation at very low supply voltage requires careful attention to ensure that the common-mode voltage remains within its linear range. See “Input Common-Mode Voltage Range.”

The INA126/2126 can be operated from a single power supply with careful attention to input common-mode range, output voltage swing of both op amps and the voltage applied to the Ref terminal. Figure 4 shows a bridge amplifier circuit operated from a single +5V power supply. The bridge provides an input common-mode voltage near 2.5V, with a relatively small differential voltage.

INPUT PROTECTION

The inputs are protected with internal diodes connected to the power supply rails. These diodes will clamp the applied signal to prevent it from exceeding the power supplies by more than approximately 0.7V. If the signal source voltage can exceed the power supplies, the source current should be limited to less than 10mA. This can generally be done with a series resistor. Some signal sources are inherently current-limited and do not require limiting resistors.

CHANNEL CROSSTALK—DUAL VERSION

The two channels of the INA2126 are completely independent, including all bias circuitry. At DC and low frequency there is virtually no signal coupling between channels. Crosstalk increases with frequency and is dependent on circuit gain, source impedance and signal characteristics.

As source impedance increases, careful circuit layout will help achieve lowest channel crosstalk. Most crosstalk is produced by capacitive coupling of signals from one channel to the input section of the other channel. To minimize coupling, separate the input traces as far as practical from any signals associated with the opposite channel. A grounded guard trace surrounding the inputs helps reduce stray coupling between channels. Carefully balance the stray capacitance of each input to ground, and run the differential inputs of each channel parallel to each other, or directly adjacent on top and bottom side of a circuit board. Stray coupling then tends to produce a common-mode signal that is rejected by the IA's input.

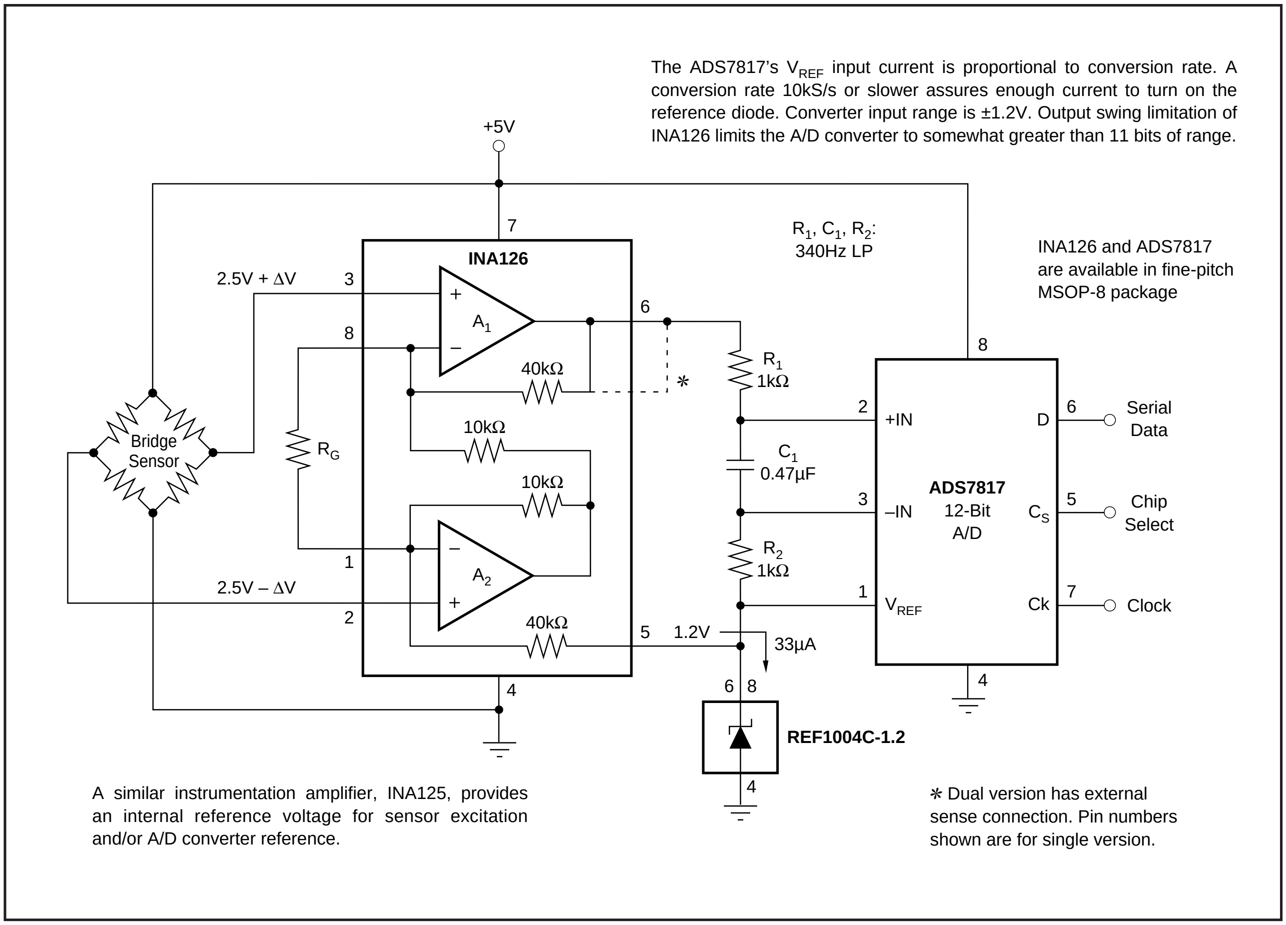
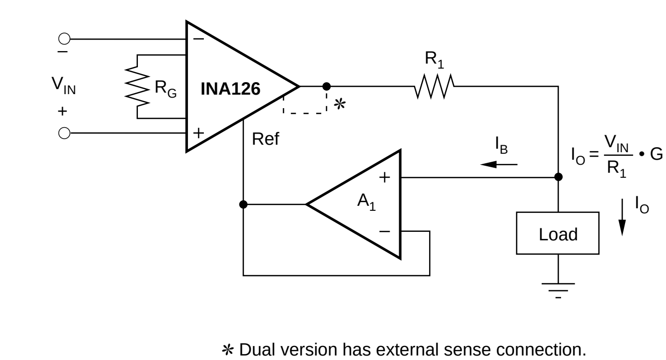


FIGURE 4. Bridge Signal Acquisition—Single 5V Supply.



A_1	I_B Error
OPA177	±1.5nA
OPA130	±20pA
OPA602	±1pA
OPA129	±100fA

FIGURE 5. Differential Voltage-to-Current Converter.

LM741 Operational Amplifier

Check for Samples: LM741

FEATURES

- **Overload Protection on the Input and Output**
- **No Latch-Up When the Common Mode Range is Exceeded**

DESCRIPTION

The LM741 series are general purpose operational amplifiers which feature improved performance over industry standards like the LM709. They are direct, plug-in replacements for the 709C, LM201, MC1439 and 748 in most applications.

The amplifiers offer many features which make their application nearly foolproof: overload protection on the input and output, no latch-up when the common mode range is exceeded, as well as freedom from oscillations.

The LM741C is identical to the LM741/LM741A except that the LM741C has their performance ensured over a 0°C to +70°C temperature range, instead of −55°C to +125°C.

Connection Diagrams

LM741H is available per JM38510/10101

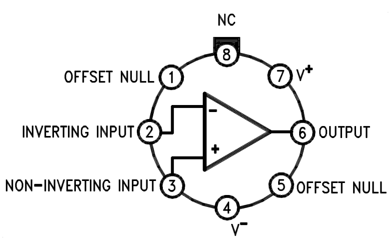


Figure 1. TO-99 Package
See Package Number LMC0008C

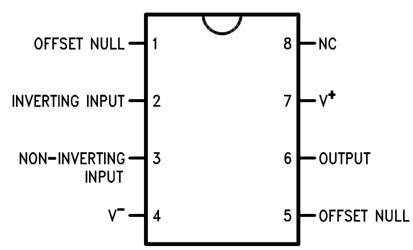


Figure 2. CDIP or PDIP Package
See Package Number NAB0008A, P0008E

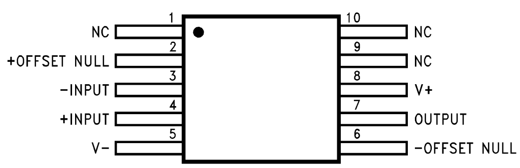


Figure 3. CLGA Package
See Package Number NAD0010A

Typical Application

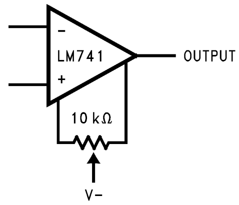


Figure 4. Offset Nulling Circuit

These devices have limited built-in ESD protection. The leads should be shorted together or the device placed in conductive foam during storage or handling to prevent electrostatic damage to the MOS gates.

Absolute Maximum Ratings[(1)(2)(3)]

	LM741A	LM741	LM741C
Supply Voltage	±22V	±22V	±18V
Power Dissipation [(4)]	500 mW	500 mW	500 mW
Differential Input Voltage	±30V	±30V	±30V
Input Voltage [(5)]	±15V	±15V	±15V
Output Short Circuit Duration	Continuous	Continuous	Continuous
Operating Temperature Range	−55°C to +125°C	−55°C to +125°C	0°C to +70°C
Storage Temperature Range	−65°C to +150°C	−65°C to +150°C	−65°C to +150°C
Junction Temperature	150°C	150°C	100°C
Soldering Information			
P0008E-Package (10 seconds)	260°C	260°C	260°C
NAB0008A- or LMC0008C-Package (10 seconds)	300°C	300°C	300°C
M-Package			
Vapor Phase (60 seconds)	215°C	215°C	215°C
Infrared (15 seconds)	215°C	215°C	215°C
ESD Tolerance [(6)]	400V	400V	400V

(1) "Absolute Maximum Ratings" indicate limits beyond which damage to the device may occur. Operating Ratings indicate conditions for which the device is functional, but do not ensure specific performance limits.
(2) For military specifications see RETS741X for LM741 and RETS741AX for LM741A.
(3) If Military/Aerospace specified devices are required, please contact the TI Sales Office/Distributors for availability and specifications.
(4) For operation at elevated temperatures, these devices must be derated based on thermal resistance, and T_j max. (listed under "Absolute Maximum Ratings"). $T_j = T_A + (\theta_{jA} P_D)$.
(5) For supply voltages less than ±15V, the absolute maximum input voltage is equal to the supply voltage.
(6) Human body model, 1.5 kΩ in series with 100 pF.

Electrical Characteristics[(1)]

Parameter	Test Conditions	LM741A			LM741			LM741C			Units
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
Input Offset Voltage	$T_A = 25°C$										
	$R_S \leq 10$ kΩ					1.0	5.0		2.0	6.0	mV
	$R_S \leq 50\Omega$		0.8	3.0							
	$T_{AMIN} \leq T_A \leq T_{AMAX}$										
	$R_S \leq 50\Omega$			4.0							mV
	$R_S \leq 10$ kΩ						6.0			7.5	
Average Input Offset Voltage Drift				15							μV/°C

(1) Unless otherwise specified, these specifications apply for V_S = ±15V, −55°C ≤ T_A ≤ +125°C (LM741/LM741A). For the LM741C/LM741E, these specifications are limited to 0°C ≤ T_A ≤ +70°C.

Electrical Characteristics[1] (continued)

Parameter	Test Conditions	LM741A			LM741			LM741C			Units
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
Input Offset Voltage Adjustment Range	T_A = 25°C, V_S = ±20V	±10				±15			±15		mV
Input Offset Current	T_A = 25°C		3.0	30		20	200		20	200	nA
	$T_{AMIN} \le T_A \le T_{AMAX}$			70		85	500			300	nA
Average Input Offset Current Drift				0.5							nA/°C
Input Bias Current	T_A = 25°C		30	80		80	500		80	500	nA
	$T_{AMIN} \le T_A \le T_{AMAX}$			0.210			1.5			0.8	μA
Input Resistance	T_A = 25°C, V_S = ±20V	1.0	6.0		0.3	2.0		0.3	2.0		MΩ
	$T_{AMIN} \le T_A \le T_{AMAX}$, V_S = ±20V	0.5									MΩ
Input Voltage Range	T_A = 25°C							±12	±13		V
	$T_{AMIN} \le T_A \le T_{AMAX}$				±12	±13					V
Large Signal Voltage Gain	T_A = 25°C, R_L ≥ 2 kΩ										V/mV
	V_S = ±20V, V_O = ±15V	50									V/mV
	V_S = ±15V, V_O = ±10V				50	200		20	200		V/mV
	$T_{AMIN} \le T_A \le T_{AMAX}$, R_L ≥ 2 kΩ,										V/mV
	V_S = ±20V, V_O = ±15V	32									V/mV
	V_S = ±15V, V_O = ±10V				25			15			V/mV
	V_S = ±5V, V_O = ±2V	10									V/mV
Output Voltage Swing	V_S = ±20V										V
	R_L ≥ 10 kΩ	±16									V
	R_L ≥ 2 kΩ	±15									V
	V_S = ±15V										V
	R_L ≥ 10 kΩ				±12	±14		±12	±14		V
	R_L ≥ 2 kΩ				±10	±13		±10	±13		V
Output Short Circuit Current	T_A = 25°C	10	25	35		25			25		mA
	$T_{AMIN} \le T_A \le T_{AMAX}$	10		40							mA
Common-Mode Rejection Ratio	$T_{AMIN} \le T_A \le T_{AMAX}$										dB
	R_S ≤ 10 kΩ, V_{CM} = ±12V				70	90		70	90		dB
	R_S ≤ 50Ω, V_{CM} = ±12V	80	95								dB
Supply Voltage Rejection Ratio	$T_{AMIN} \le T_A \le T_{AMAX}$,										dB
	V_S = ±20V to V_S = ±5V										dB
	R_S ≤ 50Ω	86	96								dB
	R_S ≤ 10 kΩ				77	96		77	96		dB
Transient Response	T_A = 25°C, Unity Gain										
Rise Time			0.25	0.8		0.3			0.3		μs
Overshoot			6.0	20		5			5		%
Bandwidth [2]	T_A = 25°C	0.437	1.5								MHz
Slew Rate	T_A = 25°C, Unity Gain	0.3	0.7			0.5			0.5		V/μs
Supply Current	T_A = 25°C					1.7	2.8		1.7	2.8	mA
Power Consumption	T_A = 25°C										mW
	V_S = ±20V		80	150							mW
	V_S = ±15V					50	85		50	85	mW

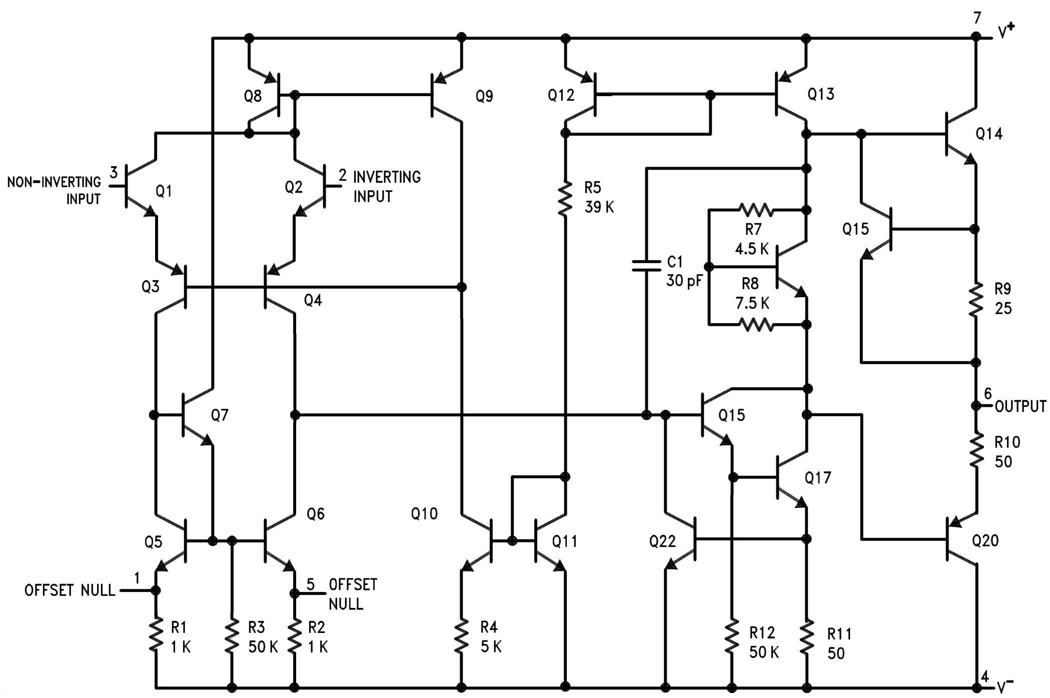
(2) Calculated value from: BW (MHz) = 0.35/Rise Time (μs).

Electrical Characteristics[1] (continued)

Parameter	Test Conditions	LM741A			LM741			LM741C			Units
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
LM741A	$V_S = \pm 20V$										
	$T_A = T_{AMIN}$			165							mW
	$T_A = T_{AMAX}$			135							
LM741	$V_S = \pm 15V$										
	$T_A = T_{AMIN}$					60	100				mW
	$T_A = T_{AMAX}$					45	75				

Thermal Resistance	CDIP (NAB0008A)	PDIP (P0008E)	TO-99 (LMC0008C)	SO-8 (M)
θ_{jA} (Junction to Ambient)	100°C/W	100°C/W	170°C/W	195°C/W
θ_{jC} (Junction to Case)	N/A	N/A	25°C/W	N/A

SCHEMATIC DIAGRAM



REVISION HISTORY

Changes from Revision B (March 2013) to Revision C	Page
• Changed layout of National Data Sheet to TI format	4

PACKAGING INFORMATION

Orderable Device	Status (1)	Package Type	Package Drawing	Pins	Package Qty	Eco Plan (2)	Lead/Ball Finish	MSL Peak Temp (3)	Op Temp (°C)	Top-Side Markings (4)	Samples
LM741CH	ACTIVE	TO-99	LMC	8	500	TBD	Call TI	Call TI	0 to 70	LM741CH	Samples
LM741CH/NOPB	ACTIVE	TO-99	LMC	8	500	Green (RoHS & no Sb/Br)	POST-PLATE	Level-1-NA-UNLIM	0 to 70	LM741CH	Samples
LM741CN	ACTIVE	PDIP	P	8	40	TBD	Call TI	Call TI	0 to 70	LM 741CN	Samples
LM741CN/NOPB	ACTIVE	PDIP	P	8	40	Green (RoHS & no Sb/Br)	SN	Level-1-NA-UNLIM	0 to 70	LM 741CN	Samples
LM741H	ACTIVE	TO-99	LMC	8	500	TBD	Call TI	Call TI	-55 to 125	LM741H	Samples
LM741H/NOPB	ACTIVE	TO-99	LMC	8	500	Green (RoHS & no Sb/Br)	POST-PLATE	Level-1-NA-UNLIM	-55 to 125	LM741H	Samples
LM741J	ACTIVE	CDIP	NAB	8	40	TBD	Call TI	Call TI	-55 to 125	LM741J	Samples
U5B7741312	ACTIVE	TO-99	LMC	8	500	TBD	Call TI	Call TI	-55 to 125	LM741H	Samples
U5B7741393	ACTIVE	TO-99	LMC	8	500	TBD	Call TI	Call TI	0 to 70	LM741CH	Samples
U9T7741393	ACTIVE	PDIP	P	8	40	TBD	Call TI	Call TI	0 to 70	LM 741CN	Samples

(1) The marketing status values are defined as follows:
ACTIVE: Product device recommended for new designs.
LIFEBUY: TI has announced that the device will be discontinued, and a lifetime-buy period is in effect.
NRND: Not recommended for new designs. Device is in production to support existing customers, but TI does not recommend using this part in a new design.
PREVIEW: Device has been announced but is not in production. Samples may or may not be available.
OBSOLETE: TI has discontinued the production of the device.

(2) Eco Plan - The planned eco-friendly classification: Pb-Free (RoHS), Pb-Free (RoHS Exempt), or Green (RoHS & no Sb/Br) - please check http://www.ti.com/productcontent for the latest availability information and additional product content details.
TBD: The Pb-Free/Green conversion plan has not been defined.
Pb-Free (RoHS): TI's terms "Lead-Free" or "Pb-Free" mean semiconductor products that are compatible with the current RoHS requirements for all 6 substances, including the requirement that lead not exceed 0.1% by weight in homogeneous materials. Where designed to be soldered at high temperatures, TI Pb-Free products are suitable for use in specified lead-free processes.
Pb-Free (RoHS Exempt): This component has a RoHS exemption for either 1) lead-based flip-chip solder bumps used between the die and package, or 2) lead-based die adhesive used between the die and leadframe. The component is otherwise considered Pb-Free (RoHS compatible) as defined above.
Green (RoHS & no Sb/Br): TI defines "Green" to mean Pb-Free (RoHS compatible), and free of Bromine (Br) and Antimony (Sb) based flame retardants (Br or Sb do not exceed 0.1% by weight in homogeneous material)

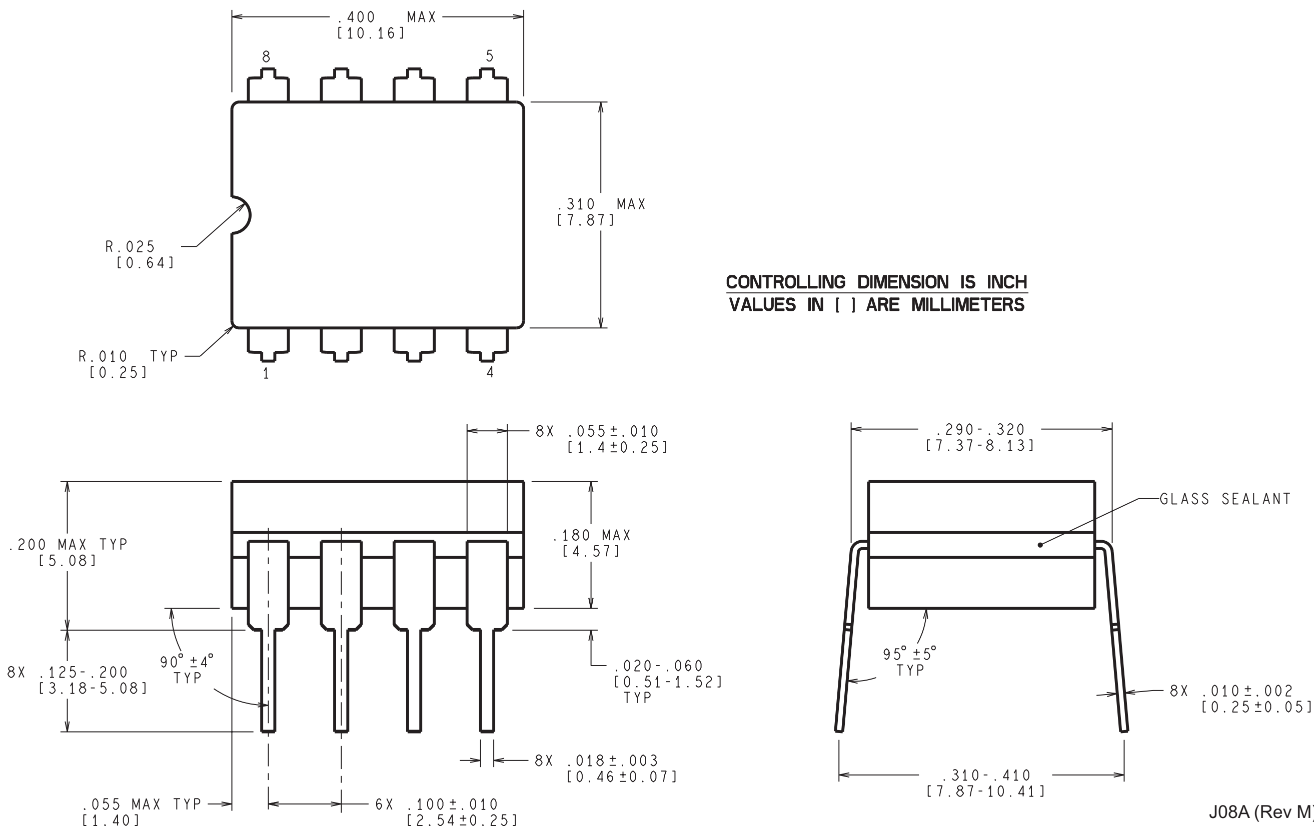
(3) MSL, Peak Temp. -- The Moisture Sensitivity Level rating according to the JEDEC industry standard classifications, and peak solder temperature.

(4) Multiple Top-Side Markings will be inside parentheses. Only one Top-Side Marking contained in parentheses and separated by a "~" will appear on a device. If a line is indented then it is a continuation of the previous line and the two combined represent the entire Top-Side Marking for that device.

Important Information and Disclaimer: The information provided on this page represents TI's knowledge and belief as of the date that it is provided. TI bases its knowledge and belief on information provided by third parties, and makes no representation or warranty as to the accuracy of such information. Efforts are underway to better integrate information from third parties. TI has taken and continues to take reasonable steps to provide representative and accurate information but may not have conducted destructive testing or chemical analysis on incoming materials and chemicals. TI and TI suppliers consider certain information to be proprietary, and thus CAS numbers and other limited information may not be available for release.

In no event shall TI's liability arising out of such information exceed the total purchase price of the TI part(s) at issue in this document sold by TI to Customer on an annual basis.

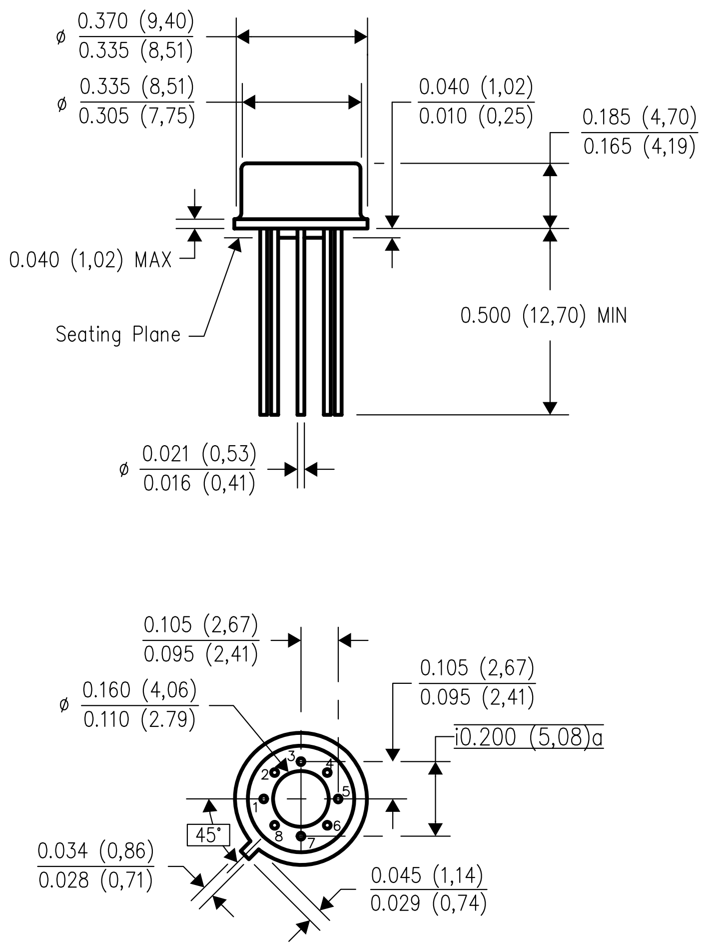
NAB0008A



CONTROLLING DIMENSION IS INCH
VALUES IN [ ] ARE MILLIMETERS

J08A (Rev M)

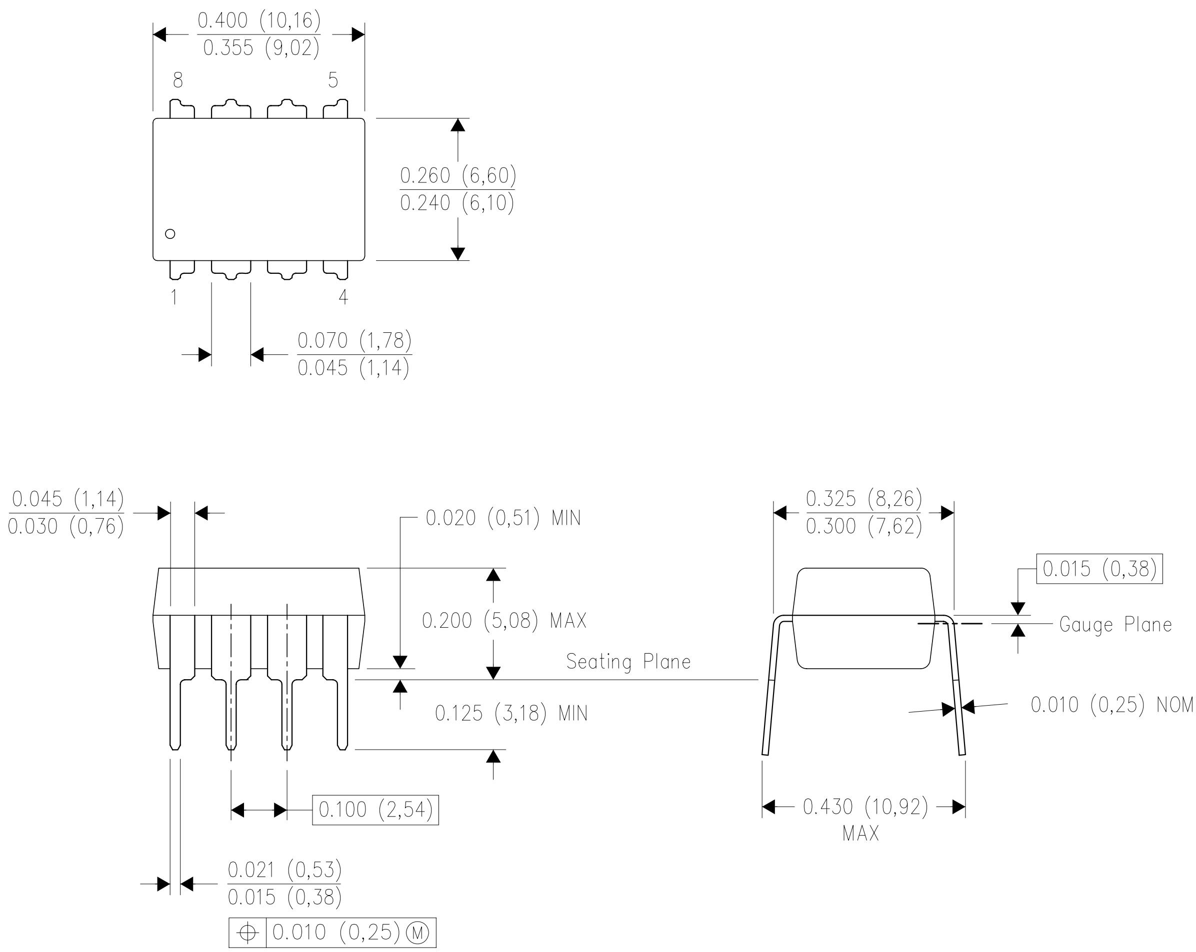
LMC (O-MBCY-W8) METAL CYLINDRICAL PACKAGE



4202483/B 09/07

NOTES:
A. All linear dimensions are in inches (millimeters).
B. This drawing is subject to change without notice.
C. Leads in true position within 0.010 (0,25) R @ MMC at seating plane.
D. Pin numbers shown for reference only. Numbers may not be marked on package.
E. Falls within JEDEC MO-002/TO-99.

P (R–PDIP–T8) PLASTIC DUAL–IN–LINE PACKAGE

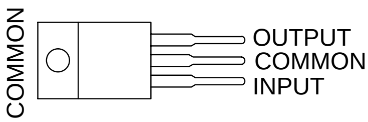


4040082/E 04/2010

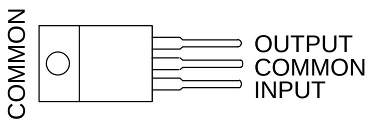
NOTES: A. All linear dimensions are in inches (millimeters).
B. This drawing is subject to change without notice.
C. Falls within JEDEC MS–001 variation BA.

- 3-Terminal Regulators
- Output Current up to 1.5 A
- Internal Thermal-Overload Protection
- High Power-Dissipation Capability
- Internal Short-Circuit Current Limiting
- Output Transistor Safe-Area Compensation

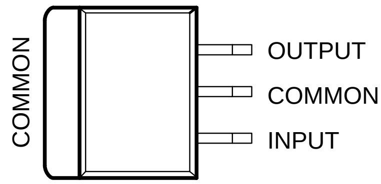
KC (TO-220) PACKAGE
(TOP VIEW)



KCS (TO-220) PACKAGE
(TOP VIEW)



KTE PACKAGE
(TOP VIEW)



description/ordering information

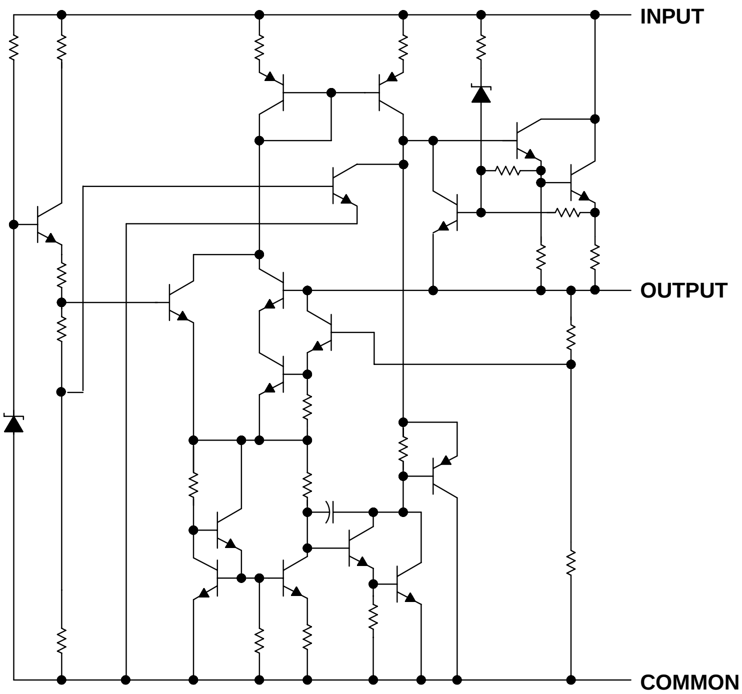
This series of fixed-voltage integrated-circuit voltage regulators is designed for a wide range of applications. These applications include on-card regulation for elimination of noise and distribution problems associated with single-point regulation. Each of these regulators can deliver up to 1.5 A of output current. The internal current-limiting and thermal-shutdown features of these regulators essentially make them immune to overload. In addition to use as fixed-voltage regulators, these devices can be used with external components to obtain adjustable output voltages and currents, and also can be used as the power-pass element in precision regulators.

ORDERING INFORMATION

T_J	$V_{O(NOM)}$ (V)	PACKAGE†		ORDERABLE PART NUMBER	TOP-SIDE MARKING
0°C to 125°C	5	POWER-FLEX (KTE)	Reel of 2000	μA7805CKTER	μA7805C
		TO-220 (KC)	Tube of 50	μA7805CKC	μA7805C
		TO-220, short shoulder (KCS)	Tube of 20	μA7805CKCS	
	8	POWER-FLEX (KTE)	Reel of 2000	μA7808CKTER	μA7808C
		TO-220 (KC)	Tube of 50	μA7808CKC	μA7808C
		TO-220, short shoulder (KCS)	Tube of 20	μA7808CKCS	
	10	POWER-FLEX (KTE)	Reel of 2000	μA7810CKTER	μA7810C
		TO-220 (KC)	Tube of 50	μA7810CKC	μA7810C
	12	POWER-FLEX (KTE)	Reel of 2000	μA7812CKTER	μA7812C
		TO-220 (KC)	Tube of 50	μA7812CKC	μA7812C
		TO-220, short shoulder (KCS)	Tube of 20	μA7812CKCS	
	15	POWER-FLEX (KTE)	Reel of 2000	μA7815CKTER	μA7815C
		TO-220 (KC)	Tube of 50	μA7815CKC	μA7815C
		TO-220, short shoulder (KCS)	Tube of 20	μA7815CKCS	
	24	POWER-FLEX (KTE)	Reel of 2000	μA7824CKTER	μA7824C
		TO-220 (KC)	Tube of 50	μA7824CKC	μA7824C

† Package drawings, standard packing quantities, thermal data, symbolization, and PCB design guidelines are available at www.ti.com/sc/package.

schematic



absolute maximum ratings over virtual junction temperature range (unless otherwise noted)†

Input voltage, V_I: μA7824C . . . . 40 V
All others . . . . 35 V
Operating virtual junction temperature, T_J . . . . 150°C
Lead temperature 1,6 mm (1/16 inch) from case for 10 seconds . . . . 260°C
Storage temperature range, T_{stg} . . . . –65°C to 150°C

† Stresses beyond those listed under “absolute maximum ratings” may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated under “recommended operating conditions” is not implied. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.

package thermal data (see Note 1)

PACKAGE	BOARD	θ_{JC}	θ_{JA}
POWER-FLEX (KTE)	High K, JESD 51-5	3°C/W	23°C/W
TO-220 (KC/KCS)	High K, JESD 51-5	3°C/W	19°C/W

NOTE 1: Maximum power dissipation is a function of T_J(max), θ_{JA}, and T_A. The maximum allowable power dissipation at any allowable ambient temperature is $P_D = (T_J(max) – T_A)/\theta_{JA}$. Operating at the absolute maximum T_J of 150°C can affect reliability.

recommended operating conditions

			MIN	MAX	UNIT
V_I	Input voltage	μA7805C	7	25	V
		μA7808C	10.5	25	
		μA7810C	12.5	28	
		μA7812C	14.5	30	
		μA7815C	17.5	30	
		μA7824C	27	38	
I_O	Output current			1.5	A
T_J	Operating virtual junction temperature	μA7800C series	0	125	°C

electrical characteristics at specified virtual junction temperature, V_I = 10 V, I_O = 500 mA (unless otherwise noted)

PARAMETER	TEST CONDITIONS	T_J†	μA7805C MIN	μA7805C TYP	μA7805C MAX	UNIT
Output voltage	I_O = 5 mA to 1 A, V_I = 7 V to 20 V, $P_D \leq$ 15 W	25°C	4.8	5	5.2	V
		0°C to 125°C	4.75		5.25	
Input voltage regulation	V_I = 7 V to 25 V	25°C		3	100	mV
	V_I = 8 V to 12 V			1	50	
Ripple rejection	V_I = 8 V to 18 V, f = 120 Hz	0°C to 125°C	62	78		dB
Output voltage regulation	I_O = 5 mA to 1.5 A	25°C		15	100	mV
	I_O = 250 mA to 750 mA			5	50	
Output resistance	f = 1 kHz	0°C to 125°C		0.017		Ω
Temperature coefficient of output voltage	I_O = 5 mA	0°C to 125°C		–1.1		mV/°C
Output noise voltage	f = 10 Hz to 100 kHz	25°C		40		μV
Dropout voltage	I_O = 1 A	25°C		2		V
Bias current		25°C		4.2	8	mA
Bias current change	V_I = 7 V to 25 V	0°C to 125°C			1.3	mA
	I_O = 5 mA to 1 A				0.5	
Short-circuit output current		25°C		750		mA
Peak output current		25°C		2.2		A

† Pulse-testing techniques maintain the junction temperature as close to the ambient temperature as possible. Thermal effects must be taken into account separately. All characteristics are measured with a 0.33-μF capacitor across the input and a 0.1-μF capacitor across the output.

electrical characteristics at specified virtual junction temperature, V_I = 14 V, I_O = 500 mA (unless otherwise noted)

PARAMETER	TEST CONDITIONS	T_J†	μA7808C MIN	μA7808C TYP	μA7808C MAX	UNIT
Output voltage	I_O = 5 mA to 1 A, V_I = 10.5 V to 23 V, $P_D \leq$ 15 W	25°C	7.7	8	8.3	V
		0°C to 125°C	7.6		8.4	
Input voltage regulation	V_I = 10.5 V to 25 V	25°C		6	160	mV
	V_I = 11 V to 17 V			2	80	
Ripple rejection	V_I = 11.5 V to 21.5 V, f = 120 Hz	0°C to 125°C	55	72		dB
Output voltage regulation	I_O = 5 mA to 1.5 A	25°C		12	160	mV
	I_O = 250 mA to 750 mA			4	80	
Output resistance	f = 1 kHz	0°C to 125°C		0.016		Ω
Temperature coefficient of output voltage	I_O = 5 mA	0°C to 125°C		–0.8		mV/°C
Output noise voltage	f = 10 Hz to 100 kHz	25°C		52		μV
Dropout voltage	I_O = 1 A	25°C		2		V
Bias current		25°C		4.3	8	mA
Bias current change	V_I = 10.5 V to 25 V	0°C to 125°C			1	mA
	I_O = 5 mA to 1 A				0.5	
Short-circuit output current		25°C		450		mA
Peak output current		25°C		2.2		A

† Pulse-testing techniques maintain the junction temperature as close to the ambient temperature as possible. Thermal effects must be taken into account separately. All characteristics are measured with a 0.33-μF capacitor across the input and a 0.1-μF capacitor across the output.

electrical characteristics at specified virtual junction temperature, V_I = 17 V, I_O = 500 mA (unless otherwise noted)

PARAMETER	TEST CONDITIONS	T_J†	μA7810C MIN	μA7810C TYP	μA7810C MAX	UNIT
Output voltage	I_O = 5 mA to 1 A, V_I = 12.5 V to 25 V, $P_D \leq$ 15 W	25°C	9.6	10	10.4	V
		0°C to 125°C	9.5	10	10.5	
Input voltage regulation	V_I = 12.5 V to 28 V	25°C		7	200	mV
	V_I = 14 V to 20 V			2	100	
Ripple rejection	V_I = 13 V to 23 V, f = 120 Hz	0°C to 125°C	55	71		dB
Output voltage regulation	I_O = 5 mA to 1.5 A	25°C		12	200	mV
	I_O = 250 mA to 750 mA			4	100	
Output resistance	f = 1 kHz	0°C to 125°C		0.018		Ω
Temperature coefficient of output voltage	I_O = 5 mA	0°C to 125°C		–1		mV/°C
Output noise voltage	f = 10 Hz to 100 kHz	25°C		70		μV
Dropout voltage	I_O = 1 A	25°C		2		V
Bias current		25°C		4.3	8	mA
Bias current change	V_I = 12.5 V to 28 V	0°C to 125°C			1	mA
	I_O = 5 mA to 1 A				0.5	
Short-circuit output current		25°C		400		mA
Peak output current		25°C		2.2		A

† Pulse-testing techniques maintain the junction temperature as close to the ambient temperature as possible. Thermal effects must be taken into account separately. All characteristics are measured with a 0.33-μF capacitor across the input and a 0.1-μF capacitor across the output.

electrical characteristics at specified virtual junction temperature, V_I = 19 V, I_O = 500 mA (unless otherwise noted)

PARAMETER	TEST CONDITIONS	T_J†	μA7812C MIN	μA7812C TYP	μA7812C MAX	UNIT
Output voltage	I_O = 5 mA to 1 A, V_I = 14.5 V to 27 V, $P_D \leq$ 15 W	25°C	11.5	12	12.5	V
		0°C to 125°C	11.4		12.6	
Input voltage regulation	V_I = 14.5 V to 30 V	25°C		10	240	mV
	V_I = 16 V to 22 V			3	120	
Ripple rejection	V_I = 15 V to 25 V, f = 120 Hz	0°C to 125°C	55	71		dB
Output voltage regulation	I_O = 5 mA to 1.5 A	25°C		12	240	mV
	I_O = 250 mA to 750 mA			4	120	
Output resistance	f = 1 kHz	0°C to 125°C		0.018		Ω
Temperature coefficient of output voltage	I_O = 5 mA	0°C to 125°C		–1		mV/°C
Output noise voltage	f = 10 Hz to 100 kHz	25°C		75		μV
Dropout voltage	I_O = 1 A	25°C		2		V
Bias current		25°C		4.3	8	mA
Bias current change	V_I = 14.5 V to 30 V	0°C to 125°C			1	mA
	I_O = 5 mA to 1 A				0.5	
Short-circuit output current		25°C		350		mA
Peak output current		25°C		2.2		A

† Pulse-testing techniques maintain the junction temperature as close to the ambient temperature as possible. Thermal effects must be taken into account separately. All characteristics are measured with a 0.33-μF capacitor across the input and a 0.1-μF capacitor across the output.

electrical characteristics at specified virtual junction temperature, V_I = 23 V, I_O = 500 mA (unless otherwise noted)

PARAMETER	TEST CONDITIONS	T_J†	μA7815C MIN	μA7815C TYP	μA7815C MAX	UNIT
Output voltage	I_O = 5 mA to 1 A, V_I = 17.5 V to 30 V, $P_D \leq$ 15 W	25°C	14.4	15	15.6	V
		0°C to 125°C	14.25		15.75	
Input voltage regulation	V_I = 17.5 V to 30 V	25°C		11	300	mV
	V_I = 20 V to 26 V			3	150	
Ripple rejection	V_I = 18.5 V to 28.5 V, f = 120 Hz	0°C to 125°C	54	70		dB
Output voltage regulation	I_O = 5 mA to 1.5 A	25°C		12	300	mV
	I_O = 250 mA to 750 mA			4	150	
Output resistance	f = 1 kHz	0°C to 125°C		0.019		Ω
Temperature coefficient of output voltage	I_O = 5 mA	0°C to 125°C		–1		mV/°C
Output noise voltage	f = 10 Hz to 100 kHz	25°C		90		μV
Dropout voltage	I_O = 1 A	25°C		2		V
Bias current		25°C		4.4	8	mA
Bias current change	V_I = 17.5 V to 30 V	0°C to 125°C			1	mA
	I_O = 5 mA to 1 A				0.5	
Short-circuit output current		25°C		230		mA
Peak output current		25°C		2.1		A

† Pulse-testing techniques maintain the junction temperature as close to the ambient temperature as possible. Thermal effects must be taken into account separately. All characteristics are measured with a 0.33-μF capacitor across the input and a 0.1-μF capacitor across the output.

electrical characteristics at specified virtual junction temperature, V_I = 33 V, I_O = 500 mA (unless otherwise noted)

PARAMETER	TEST CONDITIONS	T_J†	μA7824C MIN	μA7824C TYP	μA7824C MAX	UNIT
Output voltage	I_O = 5 mA to 1 A, V_I = 27 V to 38 V, $P_D \leq$ 15 W	25°C	23	24	25	V
		0°C to 125°C	22.8		25.2	
Input voltage regulation	V_I = 27 V to 38 V	25°C		18	480	mV
	V_I = 30 V to 36 V			6	240	
Ripple rejection	V_I = 28 V to 38 V, f = 120 Hz	0°C to 125°C	50	66		dB
Output voltage regulation	I_O = 5 mA to 1.5 A	25°C		12	480	mV
	I_O = 250 mA to 750 mA			4	240	
Output resistance	f = 1 kHz	0°C to 125°C		0.028		Ω
Temperature coefficient of output voltage	I_O = 5 mA	0°C to 125°C		–1.5		mV/°C
Output noise voltage	f = 10 Hz to 100 kHz	25°C		170		μV
Dropout voltage	I_O = 1 A	25°C		2		V
Bias current		25°C		4.6	8	mA
Bias current change	V_I = 27 V to 38 V	0°C to 125°C			1	mA
	I_O = 5 mA to 1 A				0.5	
Short-circuit output current		25°C		150		mA
Peak output current		25°C		2.1		A

† Pulse-testing techniques maintain the junction temperature as close to the ambient temperature as possible. Thermal effects must be taken into account separately. All characteristics are measured with a 0.33-μF capacitor across the input and a 0.1-μF capacitor across the output.

APPLICATION INFORMATION

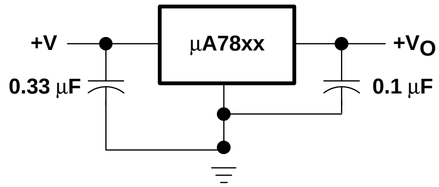


Figure 1. Fixed-Output Regulator

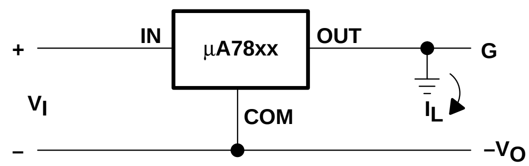
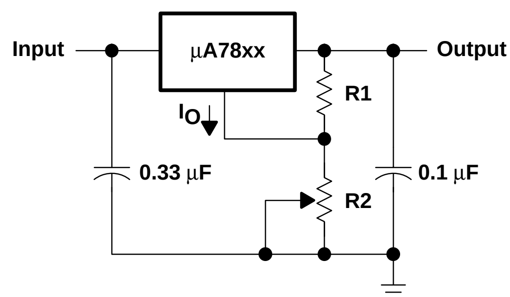


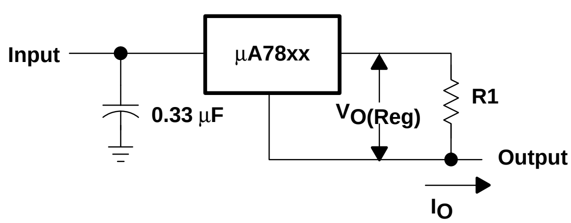
Figure 2. Positive Regulator in Negative Configuration (V_I Must Float)



NOTE A: The following formula is used when V_{XX} is the nominal output voltage (output to common) of the fixed regulator:

$$V_O = V_{xx} + \left(\frac{V_{xx}}{R1} + I_Q\right)R2$$

Figure 3. Adjustable-Output Regulator



$I_O = (V_O/R1) + I_O$ **Bias Current**

Figure 4. Current Regulator

APPLICATION INFORMATION

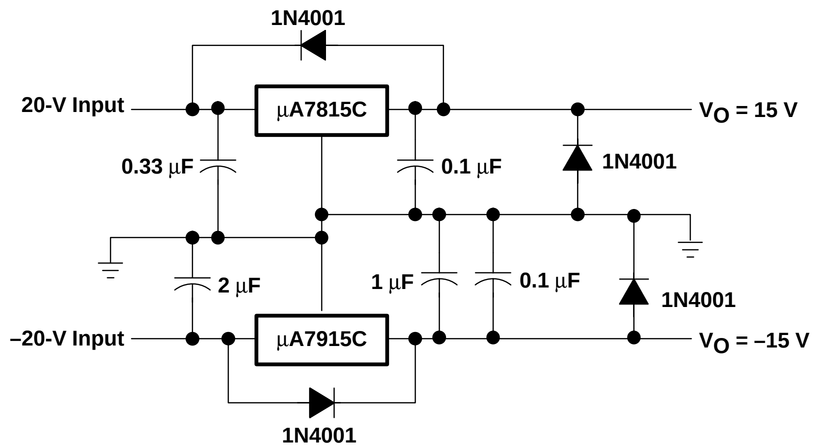


Figure 5. Regulated Dual Supply

operation with a load common to a voltage of opposite polarity

In many cases, a regulator powers a load that is not connected to ground but, instead, is connected to a voltage source of opposite polarity (e.g., operational amplifiers, level-shifting circuits, etc.). In these cases, a clamp diode should be connected to the regulator output as shown in Figure 6. This protects the regulator from output polarity reversals during startup and short-circuit operation.

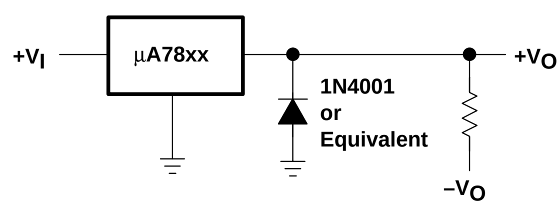


Figure 6. Output Polarity-Reversal-Protection Circuit

reverse-bias protection

Occasionally, the input voltage to the regulator can collapse faster than the output voltage. This can occur, for example, when the input supply is crowbarred during an output overvoltage condition. If the output voltage is greater than approximately 7 V, the emitter-base junction of the series-pass element (internal or external) could break down and be damaged. To prevent this, a diode shunt can be used as shown in Figure 7.

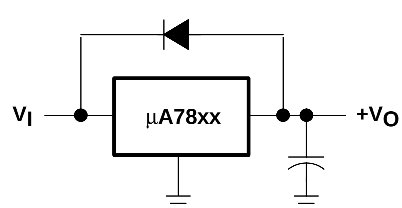
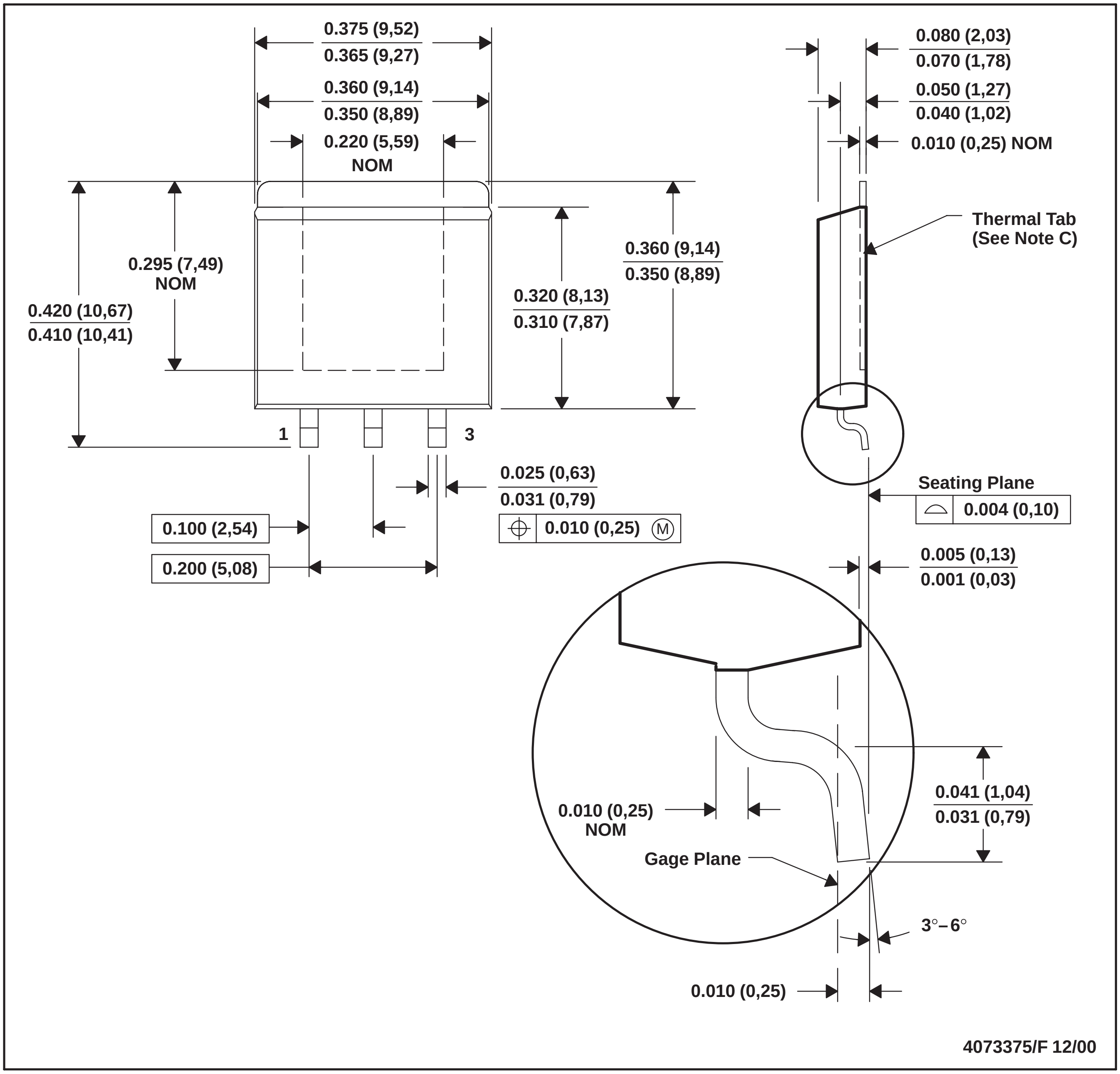


Figure 7. Reverse-Bias-Protection Circuit

MPFM001E – OCTOBER 1994 – REVISED JANUARY 2001

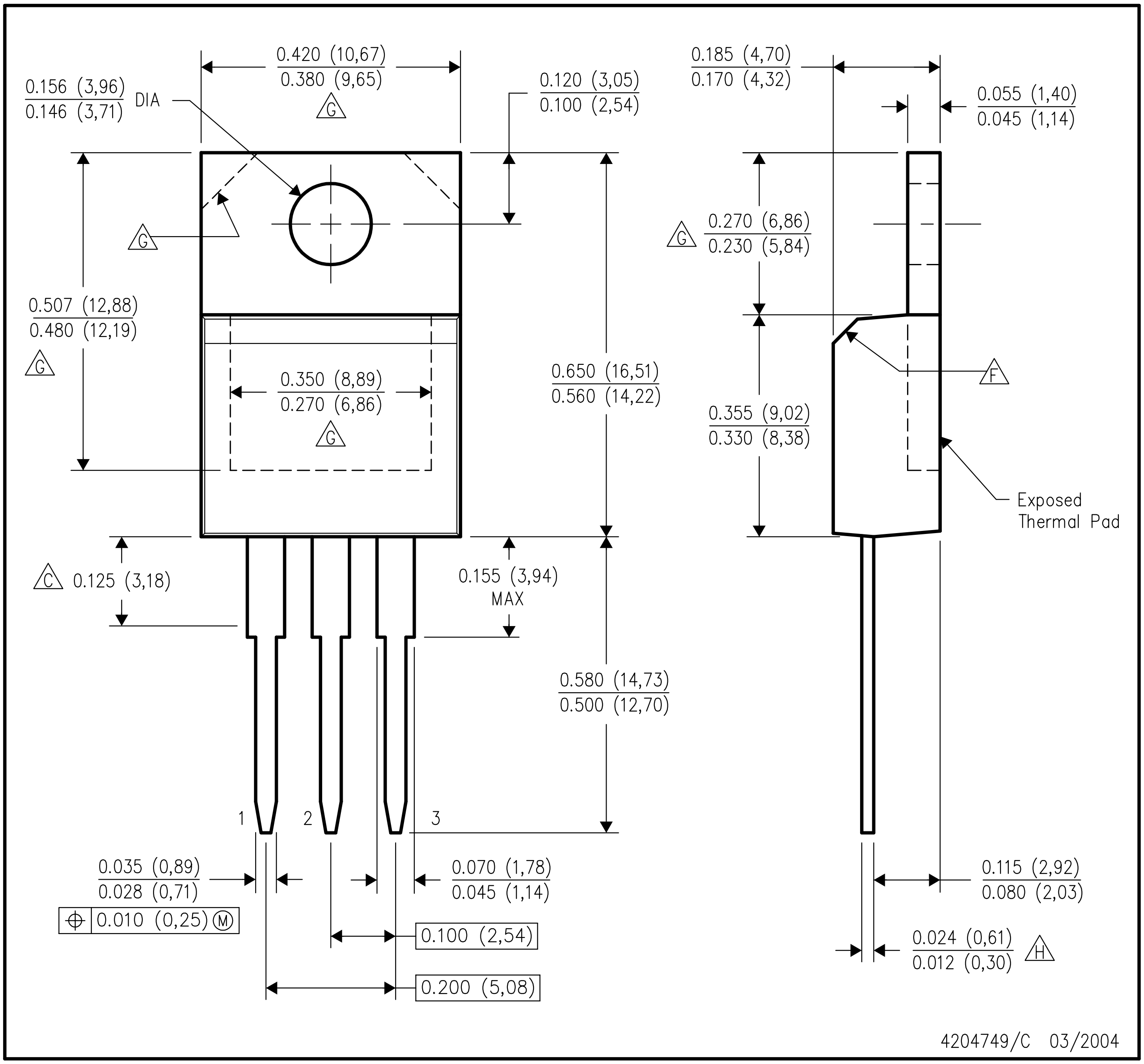
KTE (R-PSFM-G3) PowerFLEX™ PLASTIC FLANGE-MOUNT



NOTES:
- A. All linear dimensions are in inches (millimeters).
- B. This drawing is subject to change without notice.
- C. The center lead is in electrical contact with the thermal tab.
- D. Dimensions do not include mold protrusions, not to exceed 0.006 (0,15).
- E. Falls within JEDEC MO-169

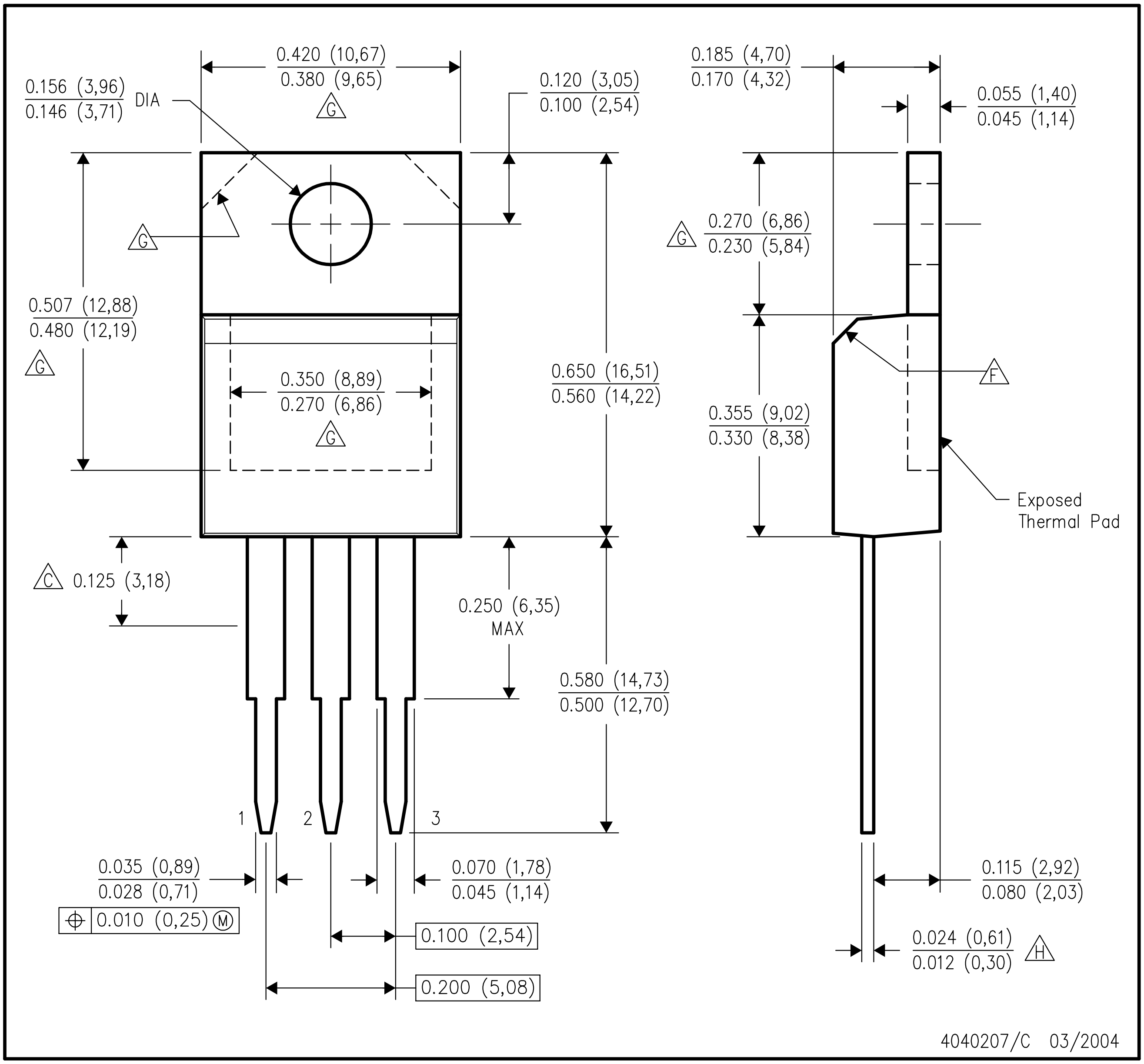
PowerFLEX is a trademark of Texas Instruments.

KCS (R–PSFM–T3) PLASTIC FLANGE–MOUNT PACKAGE



NOTES:
A. All linear dimensions are in inches (millimeters).
B. This drawing is subject to change without notice.
C. Lead dimensions are not controlled within this area.
D. All lead dimensions apply before solder dip.
E. The center lead is in electrical contact with the mounting tab.
F. The chamfer is optional.
G. Thermal pad contour optional within these dimensions.
H. Falls within JEDEC TO–220 variation AB, except minimum lead thickness.

KC (R–PSFM–T3) PLASTIC FLANGE–MOUNT PACKAGE



NOTES:
A. All linear dimensions are in inches (millimeters).
B. This drawing is subject to change without notice.
C. Lead dimensions are not controlled within this area.
D. All lead dimensions apply before solder dip.
E. The center lead is in electrical contact with the mounting tab.
F. The chamfer is optional.
G. Thermal pad contour optional within these dimensions.
H. Falls within JEDEC TO–220 variation AB, except minimum lead thickness.

IMPORTANT NOTICE

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DSP	dsp.ti.com	Broadband	www.ti.com/broadband
Interface	interface.ti.com	Digital Control	www.ti.com/digitalcontrol
Logic	logic.ti.com	Military	www.ti.com/military
Power Mgmt	power.ti.com	Optical Networking	www.ti.com/opticalnetwork
Microcontrollers	microcontroller.ti.com	Security	www.ti.com/security
		Telephony	www.ti.com/telephony
		Video & Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless

Mailing Address: Texas Instruments
Post Office Box 655303 Dallas, Texas 75265

MCP3004/3008

2.7V 4-Channel/8-Channel 10-Bit A/D Converters with SPI Serial Interface

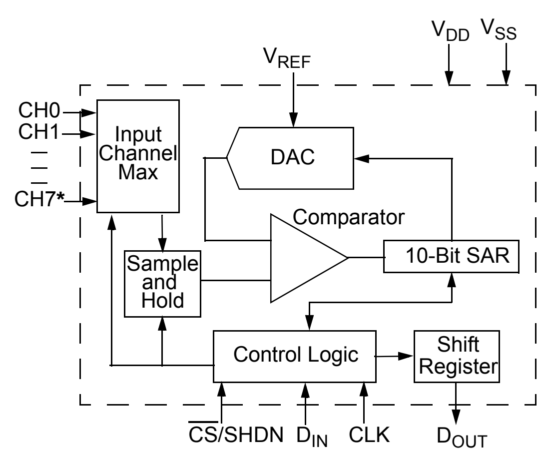
Features

- 10-bit resolution
- ± 1 LSB max DNL
- ± 1 LSB max INL
- 4 (MCP3004) or 8 (MCP3008) input channels
- Analog inputs programmable as single-ended or pseudo-differential pairs
- On-chip sample and hold
- SPI serial interface (modes 0,0 and 1,1)
- Single supply operation: 2.7V - 5.5V
- 200 ksps max. sampling rate at V_{DD} = 5V
- 75 ksps max. sampling rate at V_{DD} = 2.7V
- Low power CMOS technology
- 5 nA typical standby current, 2 µA max.
- 500 µA max. active current at 5V
- Industrial temp range: -40°C to +85°C
- Available in PDIP, SOIC and TSSOP packages

Applications

- Sensor Interface
- Process Control
- Data Acquisition
- Battery Operated Systems

Functional Block Diagram



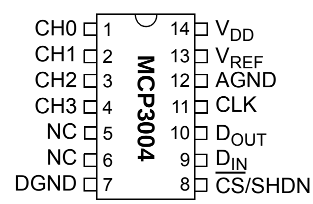
* **Note:** Channels 4-7 are available on MCP3008 Only

Description

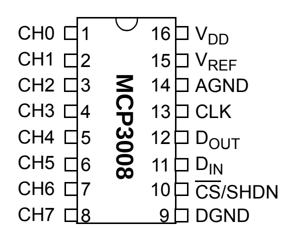
The Microchip Technology Inc. MCP3004/3008 devices are successive approximation 10-bit Analog-to-Digital (A/D) converters with on-board sample and hold circuitry. The MCP3004 is programmable to provide two pseudo-differential input pairs or four single-ended inputs. The MCP3008 is programmable to provide four pseudo-differential input pairs or eight single-ended inputs. Differential Nonlinearity (DNL) and Integral Nonlinearity (INL) are specified at ±1 LSB. Communication with the devices is accomplished using a simple serial interface compatible with the SPI protocol. The devices are capable of conversion rates of up to 200 ksps. The MCP3004/3008 devices operate over a broad voltage range (2.7V - 5.5V). Low-current design permits operation with typical standby currents of only 5 nA and typical active currents of 320 µA. The MCP3004 is offered in 14-pin PDIP, 150 mil SOIC and TSSOP packages, while the MCP3008 is offered in 16-pin PDIP and SOIC packages.

Package Types

PDIP, SOIC, TSSOP



PDIP, SOIC



NOTES:

1.0 ELECTRICAL CHARACTERISTICS

Absolute Maximum Ratings †

V_{DD} 7.0V
All Inputs and Outputs w.r.t. V_{SS} ....– 0.6V to V_{DD} + 0.6V
Storage Temperature ....–65°C to +150°C
Ambient temperature with power applied.......–65°C to +150°C
Soldering temperature of leads (10 seconds) .... +300°C
ESD Protection On All Pins (HBM) ....≥ 4 kV

† Notice: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at those or any other conditions above those indicated in the operational listings of this specification is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

ELECTRICAL SPECIFICATIONS

Electrical Characteristics: Unless otherwise noted, all parameters apply at V_{DD} = 5V, V_{REF} = 5V, T_A = -40°C to +85°C, f_{SAMPLE} = 200 ksps and f_{CLK} = 18*f_{SAMPLE}. Unless otherwise noted, typical values apply for V_{DD} = 5V, T_A = +25°C.

Parameter	Sym	Min	Typ	Max	Units	Conditions
Conversion Rate						
Conversion Time	t_{CONV}	—	—	10	clock cycles	
Analog Input Sample Time	t_{SAMPLE}		1.5		clock cycles	
Throughput Rate	f_{SAMPLE}	—	—	200 75	ksps ksps	V_{DD} = V_{REF} = 5V V_{DD} = V_{REF} = 2.7V
DC Accuracy						
Resolution			10		bits	
Integral Nonlinearity	INL	—	±0.5	±1	LSB	
Differential Nonlinearity	DNL	—	±0.25	±1	LSB	No missing codes over temperature
Offset Error		—	—	±1.5	LSB	
Gain Error		—	—	±1.0	LSB	
Dynamic Performance						
Total Harmonic Distortion		—	-76		dB	V_{IN} = 0.1V to 4.9V@1 kHz
Signal-to-Noise and Distortion (SINAD)		—	61		dB	V_{IN} = 0.1V to 4.9V@1 kHz
Spurious Free Dynamic Range		—	78		dB	V_{IN} = 0.1V to 4.9V@1 kHz
Reference Input						
Voltage Range		0.25	—	V_{DD}	V	Note 2
Current Drain		—	100 0.001	150 3	μA μA	 $\overline{CS}$ = V_{DD} = 5V
Analog Inputs						
Input Voltage Range for CH0 or CH1 in Single-Ended Mode		V_{SS}	—	V_{REF}	V	
Input Voltage Range for IN+ in pseudo-differential mode		IN-	—	V_{REF}+IN-		
Input Voltage Range for IN- in pseudo-differential mode		V_{SS}-100	—	V_{SS}+100	mV	

Note 1: This parameter is established by characterization and not 100% tested.
2: See graphs that relate linearity performance to V_{REF} levels.
3: Because the sample cap will eventually lose charge, effective clock rates below 10 kHz can affect linearity performance, especially at elevated temperatures. See **Section 6.2 "Maintaining Minimum Clock Speed"**, "Maintaining Minimum Clock Speed", for more information.

ELECTRICAL SPECIFICATIONS (CONTINUED)

Electrical Characteristics: Unless otherwise noted, all parameters apply at V_{DD} = 5V, V_{REF} = 5V, T_A = -40°C to +85°C, f_{SAMPLE} = 200 ksps and f_{CLK} = 18*f_{SAMPLE}. Unless otherwise noted, typical values apply for V_{DD} = 5V, T_A = +25°C.

Parameter	Sym	Min	Typ	Max	Units	Conditions
Leakage Current		—	0.001	±1	µA	
Switch Resistance		—	1000	—	Ω	See Figure 4-1
Sample Capacitor		—	20	—	pF	See Figure 4-1
Digital Input/Output						
Data Coding Format			Straight Binary			
High Level Input Voltage	V_{IH}	0.7 V_{DD}	—	—	V	
Low Level Input Voltage	V_{IL}		—	0.3 V_{DD}	V	
High Level Output Voltage	V_{OH}	4.1	—	—	V	I_{OH} = -1 mA, V_{DD} = 4.5V
Low Level Output Voltage	V_{OL}	—	—	0.4	V	I_{OL} = 1 mA, V_{DD} = 4.5V
Input Leakage Current	I_{LI}	-10	—	10	µA	V_{IN} = V_{SS} or V_{DD}
Output Leakage Current	I_{LO}	-10	—	10	µA	V_{OUT} = V_{SS} or V_{DD}
Pin Capacitance (All Inputs/Outputs)	C_{IN}, C_{OUT}	—	—	10	pF	V_{DD} = 5.0V (**Note 1**) T_A = 25°C, f = 1 MHz
Timing Parameters						
Clock Frequency	f_{CLK}	—	—	3.6 1.35	MHz MHz	V_{DD} = 5V (**Note 3**) V_{DD} = 2.7V (**Note 3**)
Clock High Time	t_{HI}	125	—	—	ns	
Clock Low Time	t_{LO}	125	—	—	ns	
$\overline{CS}$ Fall To First Rising CLK Edge	t_{SUCS}	100	—	—	ns	
$\overline{CS}$ Fall To Falling CLK Edge	t_{CSD}	—	—	0	ns	
Data Input Setup Time	t_{SU}	50	—	—	ns	
Data Input Hold Time	t_{HD}	50	—	—	ns	
CLK Fall To Output Data Valid	t_{DO}	—	—	125 200	ns ns	V_{DD} = 5V, See Figure 1-2 V_{DD} = 2.7V, See Figure 1-2
CLK Fall To Output Enable	t_{EN}	—	—	125 200	ns ns	V_{DD} = 5V, See Figure 1-2 V_{DD} = 2.7V, See Figure 1-2
$\overline{CS}$ Rise To Output Disable	t_{DIS}	—	—	100	ns	See Test Circuits, Figure 1-2
$\overline{CS}$ Disable Time	t_{CSH}	270	—	—	ns	
D_{OUT} Rise Time	t_R	—	—	100	ns	See Test Circuits, Figure 1-2 (**Note 1**)
D_{OUT} Fall Time	t_F	—	—	100	ns	See Test Circuits, Figure 1-2 (**Note 1**)

Note 1: This parameter is established by characterization and not 100% tested.

2: See graphs that relate linearity performance to V_{REF} levels.

3: Because the sample cap will eventually lose charge, effective clock rates below 10 kHz can affect linearity performance, especially at elevated temperatures. See **Section 6.2 "Maintaining Minimum Clock Speed"**, "Maintaining Minimum Clock Speed", for more information.

ELECTRICAL SPECIFICATIONS (CONTINUED)

Parameter	Sym	Min	Typ	Max	Units	Conditions
Electrical Characteristics: Unless otherwise noted, all parameters apply at V_{DD} = 5V, V_{REF} = 5V, T_A = -40°C to +85°C, f_{SAMPLE} = 200 ksps and f_{CLK} = 18*f_{SAMPLE}. Unless otherwise noted, typical values apply for V_{DD} = 5V, T_A = +25°C.						
Power Requirements						
Operating Voltage	V_{DD}	2.7	—	5.5	V	
Operating Current	I_{DD}	—	425 225	550	µA	V_{DD} = V_{REF} = 5V, D_{OUT} unloaded V_{DD} = V_{REF} = 2.7V, D_{OUT} unloaded
Standby Current	I_{DDS}	—	0.005	2	µA	$\overline{CS}$ = V_{DD} = 5.0V

Note 1: This parameter is established by characterization and not 100% tested.
2: See graphs that relate linearity performance to V_{REF} levels.
3: Because the sample cap will eventually lose charge, effective clock rates below 10 kHz can affect linearity performance, especially at elevated temperatures. See **Section 6.2 "Maintaining Minimum Clock Speed"**, "Maintaining Minimum Clock Speed", for more information.

TEMPERATURE CHARACTERISTICS

Parameters	Sym	Min	Typ	Max	Units	Conditions
Electrical Specifications: Unless otherwise indicated, V_{DD} = +2.7V to +5.5V, V_{SS} = GND.						
Temperature Ranges						
Specified Temperature Range	T_A	-40	—	+85	°C	
Operating Temperature Range	T_A	-40	—	+85	°C	
Storage Temperature Range	T_A	-65	—	+150	°C	
Thermal Package Resistances						
Thermal Resistance, 14L-PDIP	θ_{JA}	—	70	—	°C/W	
Thermal Resistance, 14L-SOIC	θ_{JA}	—	108	—	°C/W	
Thermal Resistance, 14L-TSSOP	θ_{JA}	—	100	—	°C/W	
Thermal Resistance, 16L-PDIP	θ_{JA}	—	70	—	°C/W	
Thermal Resistance, 16L-SOIC	θ_{JA}	—	90	—	°C/W	

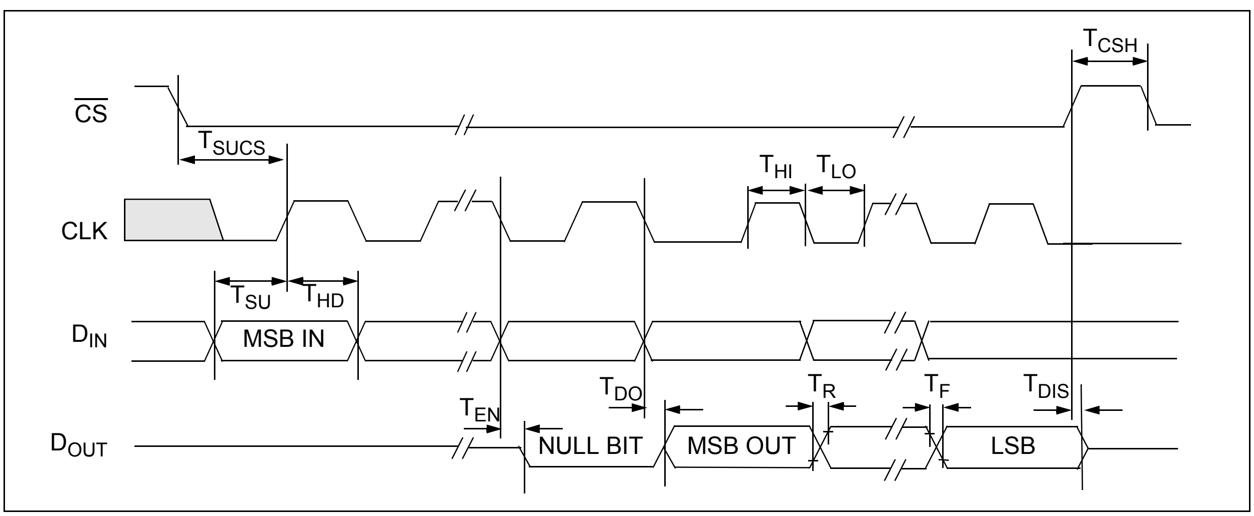


FIGURE 1-1: *Serial Interface Timing.*

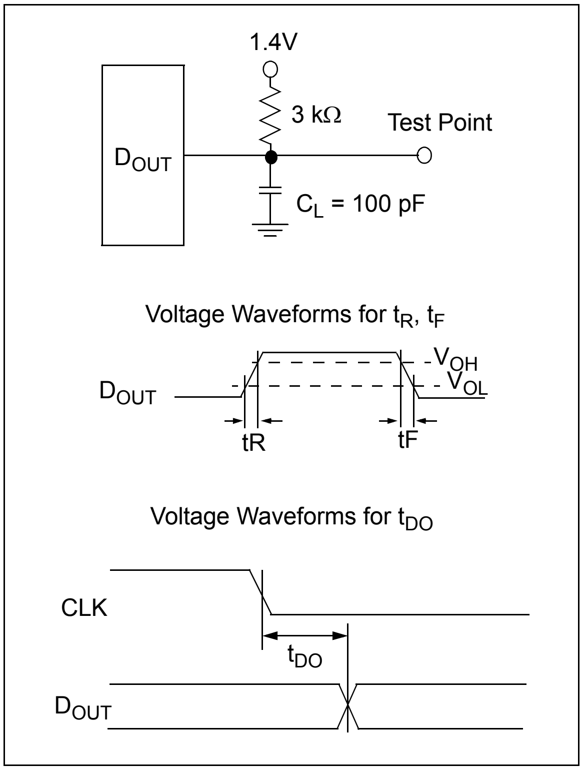


FIGURE 1-2: *Load Circuit for t_R, t_F, t_{DO}.*

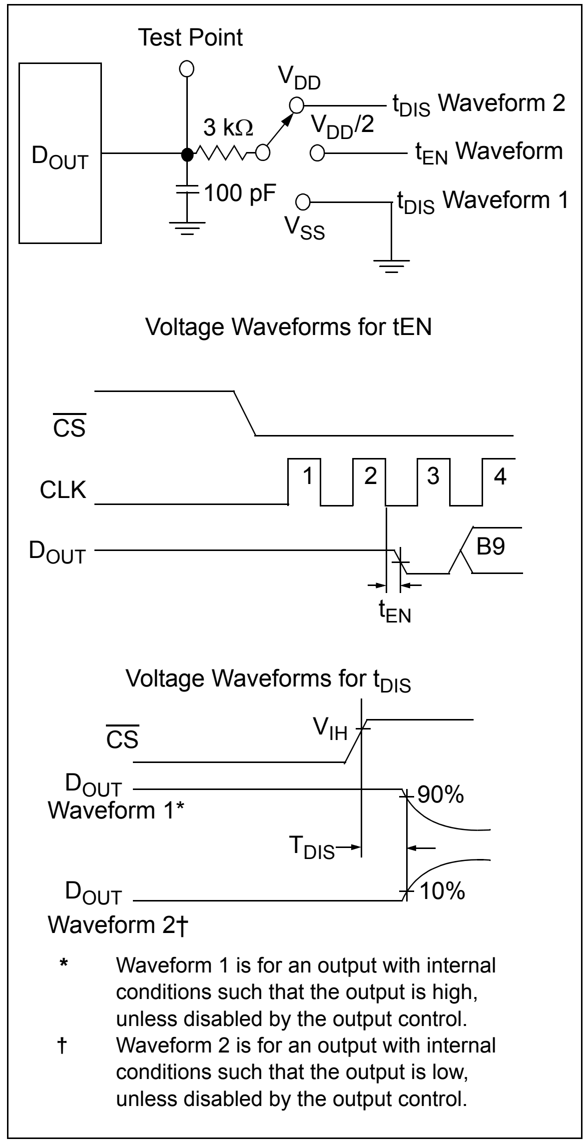


FIGURE 1-3: *Load circuit for t_{DIS} and t_{EN}.*

2.0 TYPICAL PERFORMANCE CHARACTERISTICS

Note: The graphs and tables provided following this note are a statistical summary based on a limited number of samples and are provided for informational purposes only. The performance characteristics listed herein are not tested or guaranteed. In some graphs or tables, the data presented may be outside the specified operating range (e.g., outside specified power supply range) and therefore outside the warranted range.

Note: Unless otherwise indicated, $V_{DD} = V_{REF} = 5V$, $f_{CLK} = 18^* f_{SAMPLE}$, $T_A = +25°C$.

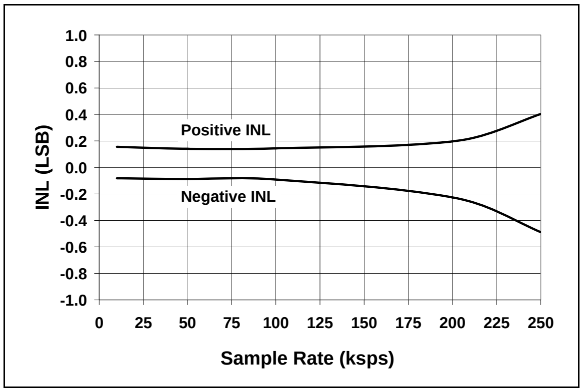


FIGURE 2-1: *Integral Nonlinearity (INL) vs. Sample Rate.*

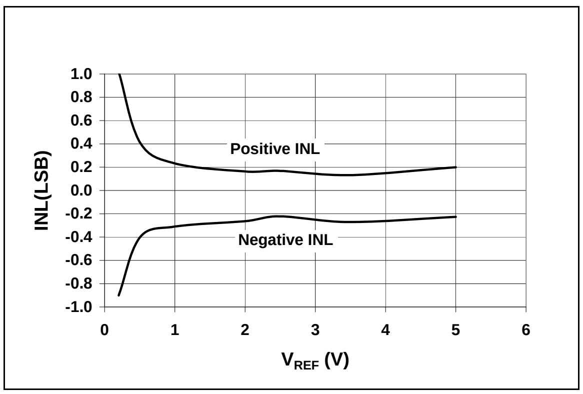


FIGURE 2-2: *Integral Nonlinearity (INL) vs. V_{REF}*

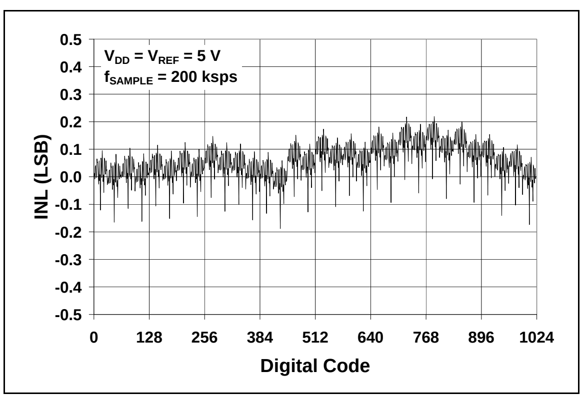


FIGURE 2-3: *Integral Nonlinearity (INL) vs. Code (Representative Part).*

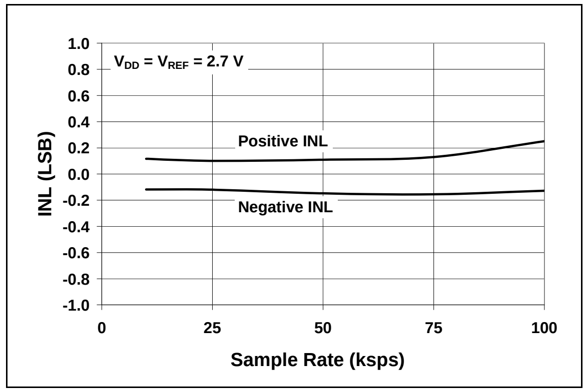


FIGURE 2-4: *Integral Nonlinearity (INL) vs. Sample Rate (V_{DD} = 2.7V).*

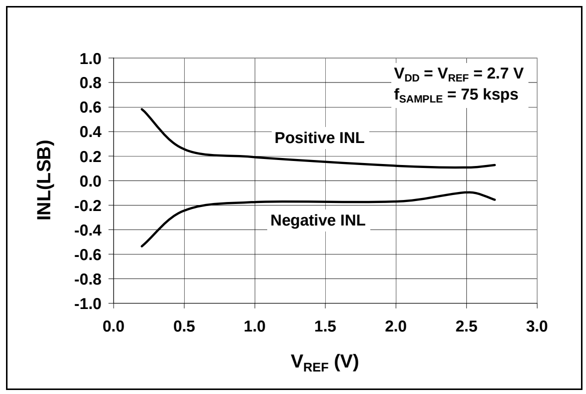


FIGURE 2-5: *Integral Nonlinearity (INL) vs. V_{REF} (V_{DD} = 2.7V).*

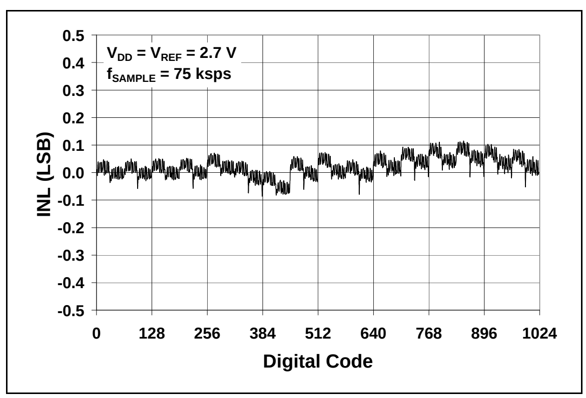


FIGURE 2-6: *Integral Nonlinearity (INL) vs. Code (Representative Part, V_{DD} = 2.7V).*

Note: Unless otherwise indicated, $V_{DD} = V_{REF} = 5V$, $f_{CLK} = 18^* f_{SAMPLE}$, $T_A = +25°C$.

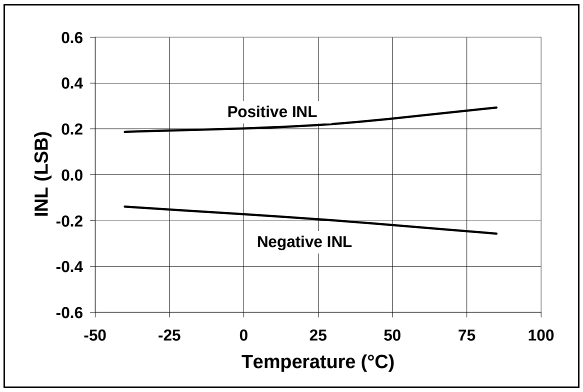


FIGURE 2-7: *Integral Nonlinearity (INL) vs. Temperature.*

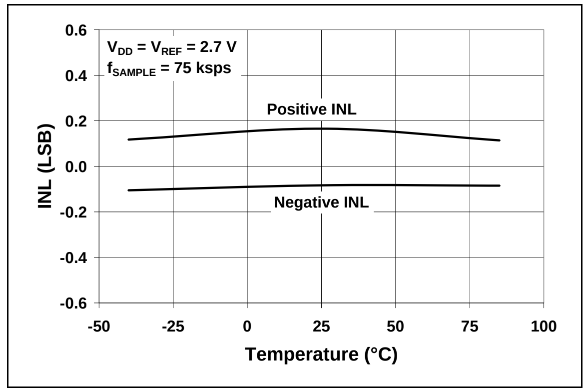


FIGURE 2-10: *Integral Nonlinearity (INL) vs. Temperature (V_{DD} = 2.7V).*

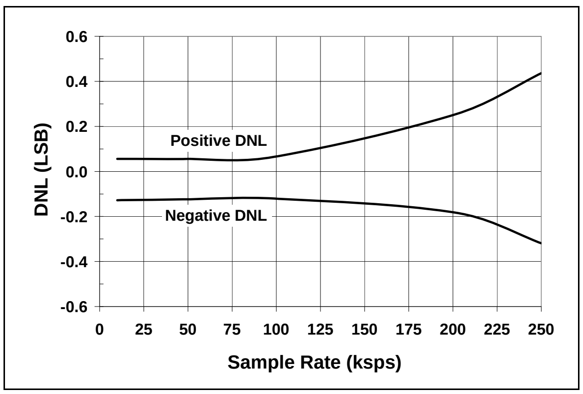


FIGURE 2-8: *Differential Nonlinearity (DNL) vs. Sample Rate.*

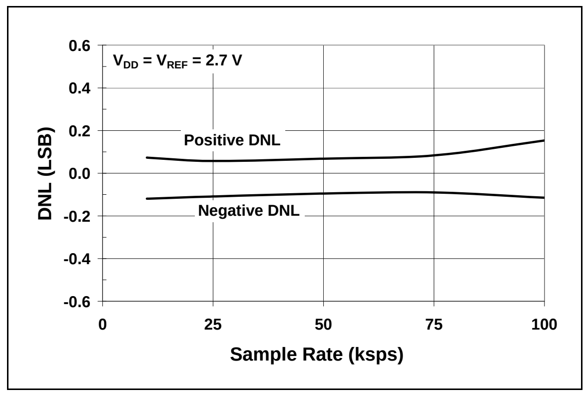


FIGURE 2-11: *Differential Nonlinearity (DNL) vs. Sample Rate (V_{DD} = 2.7V).*

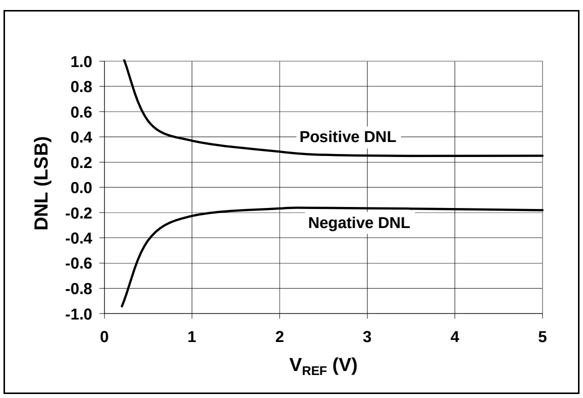


FIGURE 2-9: *Differential Nonlinearity (DNL) vs. V_{REF}*

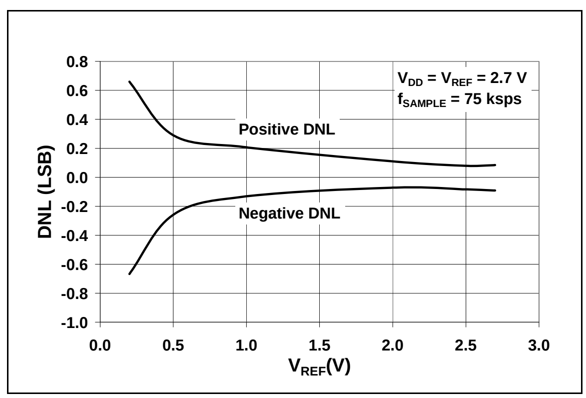


FIGURE 2-12: *Differential Nonlinearity (DNL) vs. V_{REF} (V_{DD} = 2.7V).*

Note: Unless otherwise indicated, $V_{DD} = V_{REF} = 5V$, $f_{CLK} = 18^* f_{SAMPLE}$, $T_A = +25°C$.

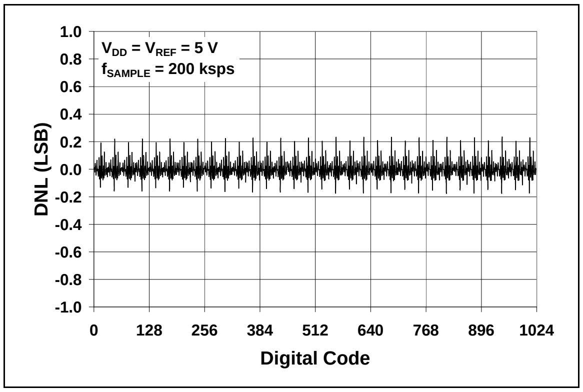


FIGURE 2-13: *Differential Nonlinearity (DNL) vs. Code (Representative Part).*

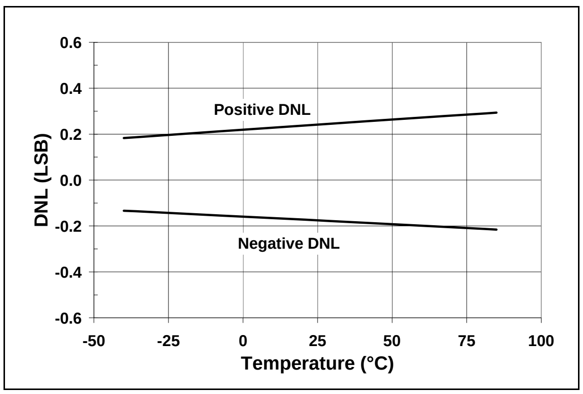


FIGURE 2-14: *Differential Nonlinearity (DNL) vs. Temperature.*

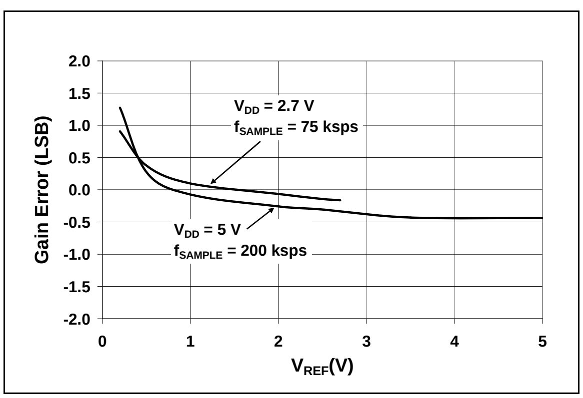


FIGURE 2-15: *Gain Error vs. V_{REF}*

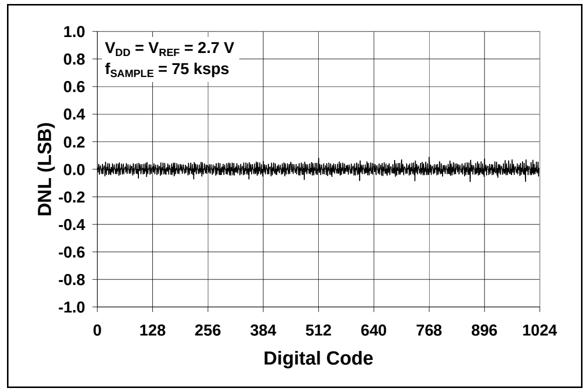


FIGURE 2-16: *Differential Nonlinearity (DNL) vs. Code (Representative Part, V_{DD} =2.7V).*

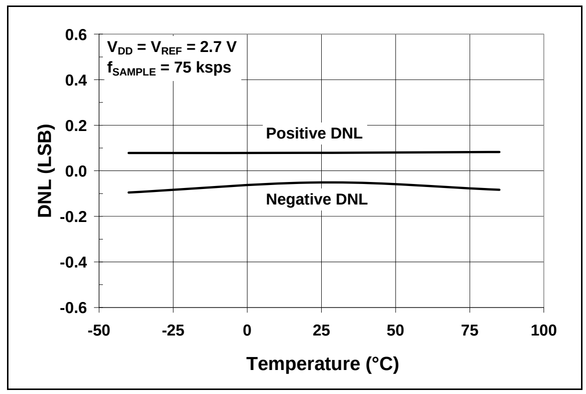


FIGURE 2-17: *Differential Nonlinearity (DNL) vs. Temperature (V_{DD} = 2.7V).*

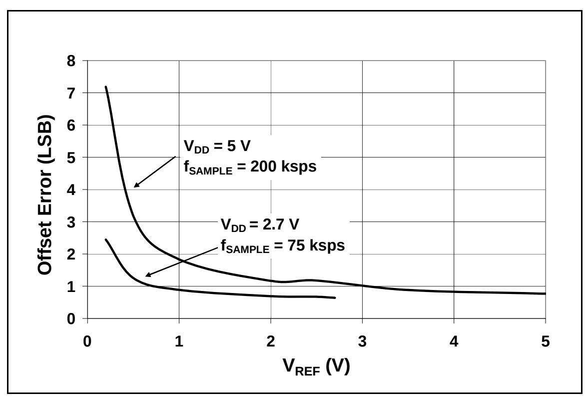


FIGURE 2-18: *Offset Error vs. V_{REF}*

Note: Unless otherwise indicated, $V_{DD} = V_{REF} = 5V$, $f_{CLK} = 18^* f_{SAMPLE}$, $T_A = +25°C$.

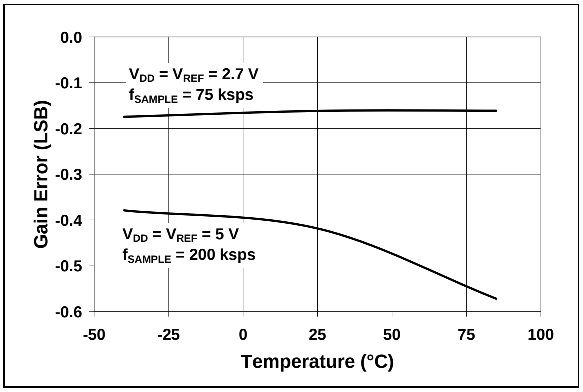


FIGURE 2-19: *Gain Error vs. Temperature.*

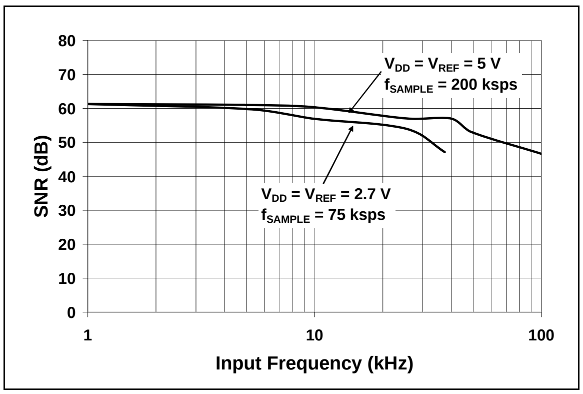


FIGURE 2-20: *Signal-to-Noise (SNR) vs. Input Frequency.*

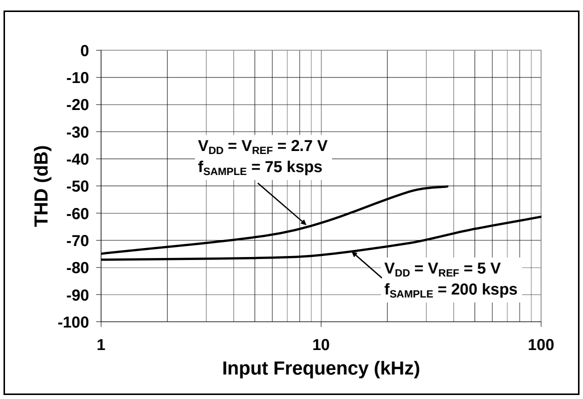


FIGURE 2-21: *Total Harmonic Distortion (THD) vs. Input Frequency.*

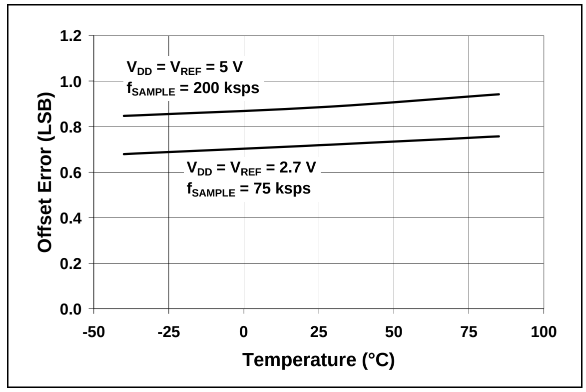


FIGURE 2-22: *Offset Error vs. Temperature.*

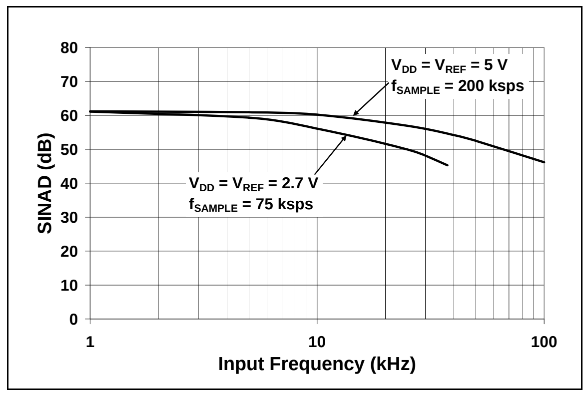


FIGURE 2-23: *Signal-to-Noise and Distortion (SINAD) vs. Input Frequency.*

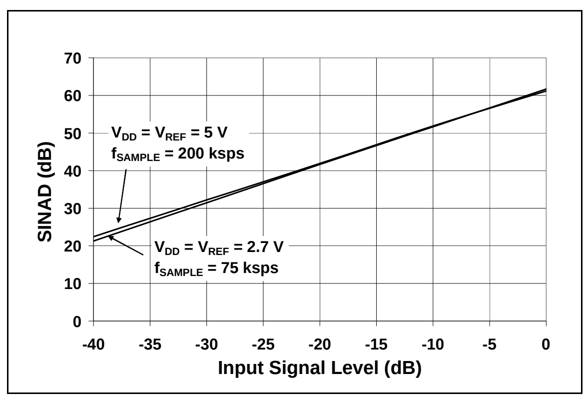


FIGURE 2-24: *Signal-to-Noise and Distortion (SINAD) vs. Input Signal Level.*

Note: Unless otherwise indicated, $V_{DD} = V_{REF} = 5V$, $f_{CLK} = 18^* f_{SAMPLE}$, $T_A = +25°C$.

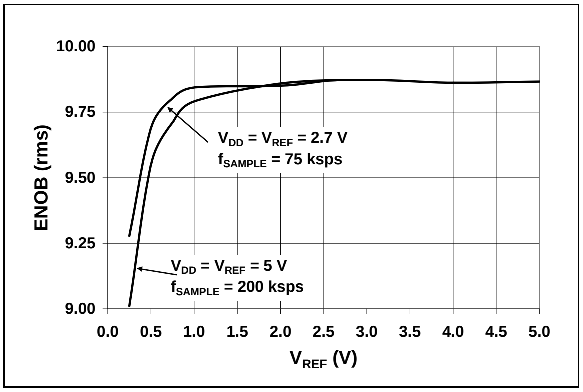


FIGURE 2-25: *Effective Number of Bits (ENOB) vs. V_{REF}*

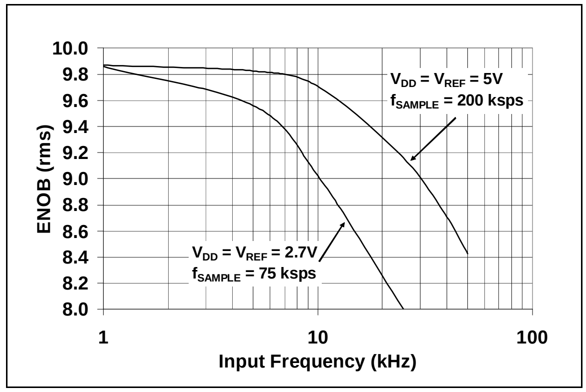


FIGURE 2-28: *Effective Number of Bits (ENOB) vs. Input Frequency.*

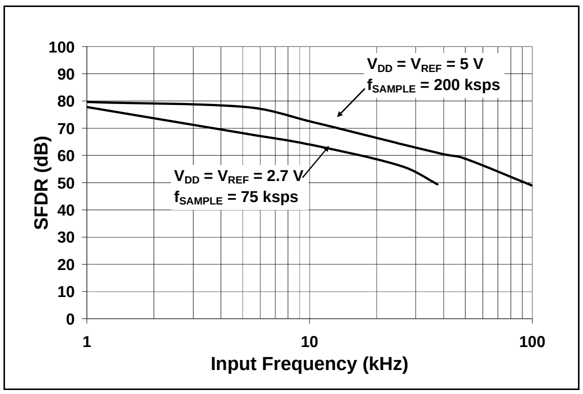


FIGURE 2-26: *Spurious Free Dynamic Range (SFDR) vs. Input Frequency.*

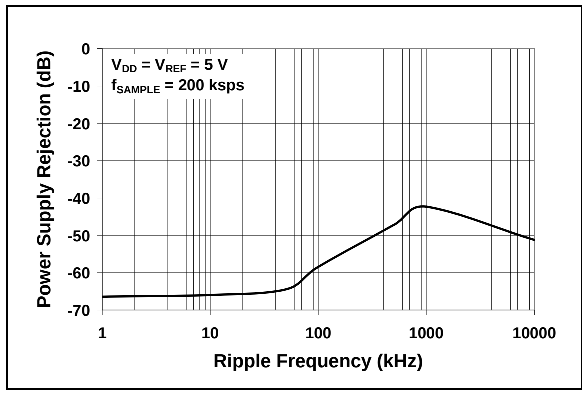


FIGURE 2-29: *Power Supply Rejection (PSR) vs. Ripple Frequency.*

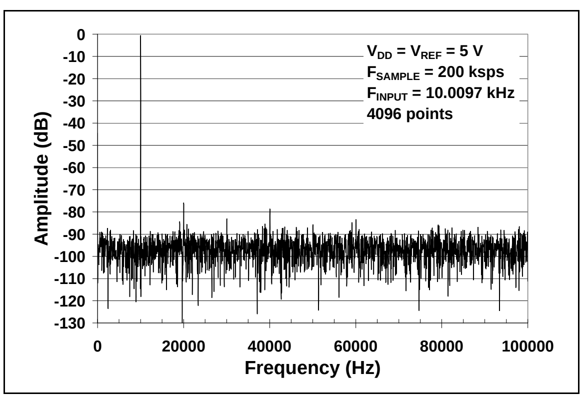


FIGURE 2-27: *Frequency Spectrum of 10 kHz Input (Representative Part).*

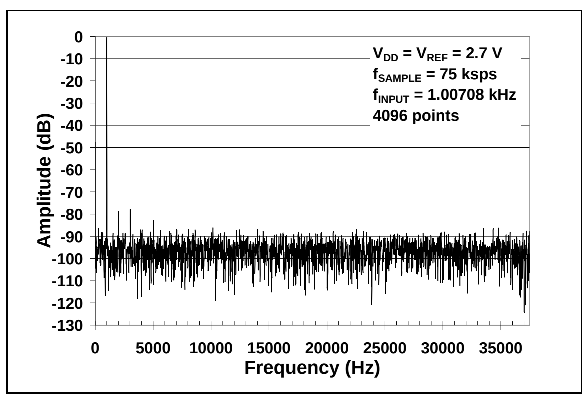


FIGURE 2-30: *Frequency Spectrum of 1 kHz Input (Representative Part, V_{DD} = 2.7V).*

Note: Unless otherwise indicated, $V_{DD} = V_{REF} = 5V$, $f_{CLK} = 18^* f_{SAMPLE}$, $T_A = +25°C$.

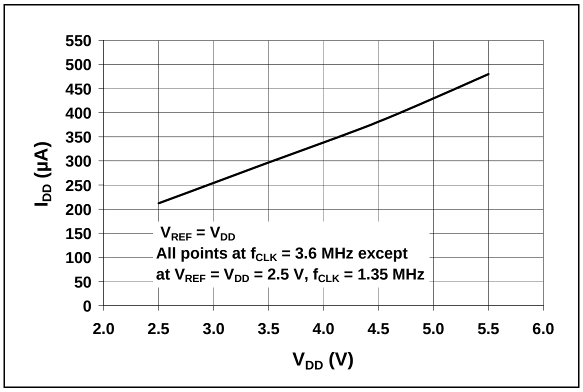


FIGURE 2-31: *I_{DD} vs. V_{DD}.*

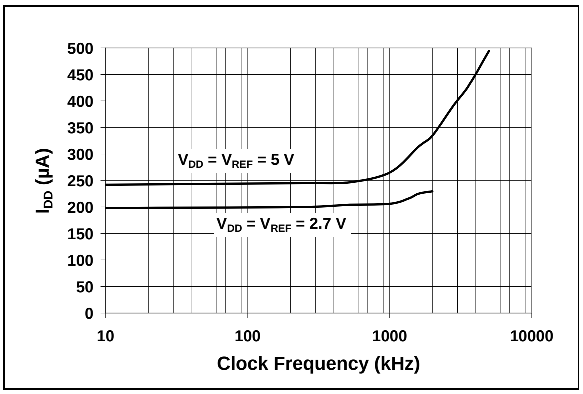


FIGURE 2-32: *I_{DD} vs. Clock Frequency.*

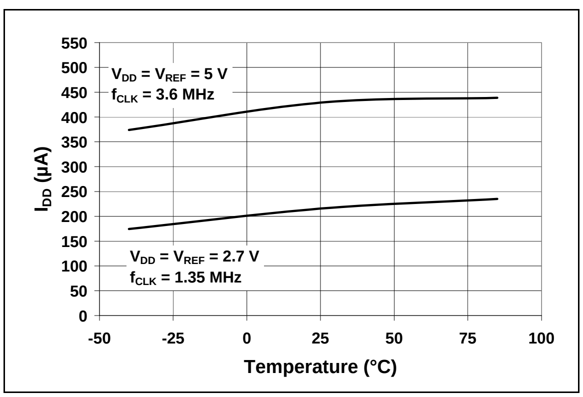


FIGURE 2-33: *I_{DD} vs. Temperature.*

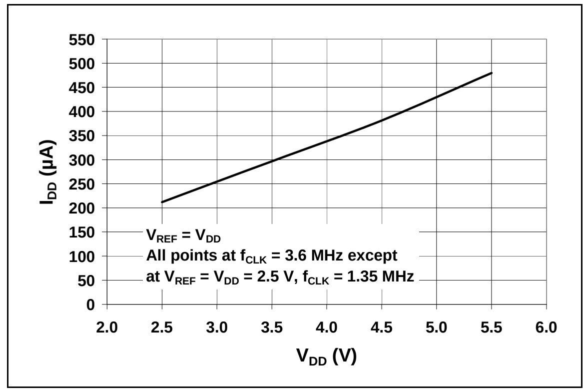


FIGURE 2-34: *I_{REF} vs. V_{DD}.*

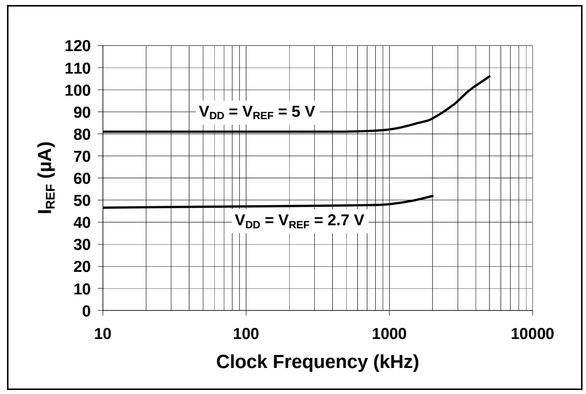


FIGURE 2-35: *I_{REF} vs. Clock Frequency.*

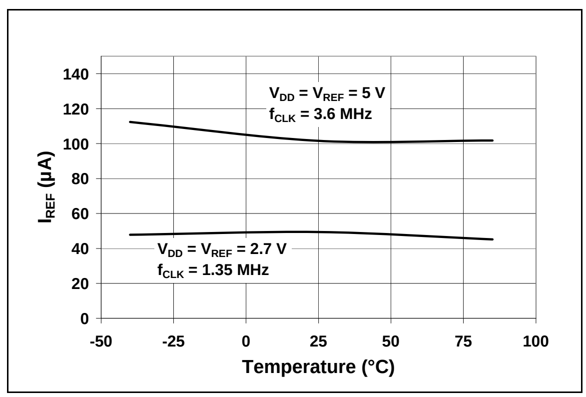


FIGURE 2-36: *I_{REF} vs. Temperature.*

Note: Unless otherwise indicated, $V_{DD} = V_{REF} = 5V$, $f_{CLK} = 18^* f_{SAMPLE}$, $T_A = +25°C$.

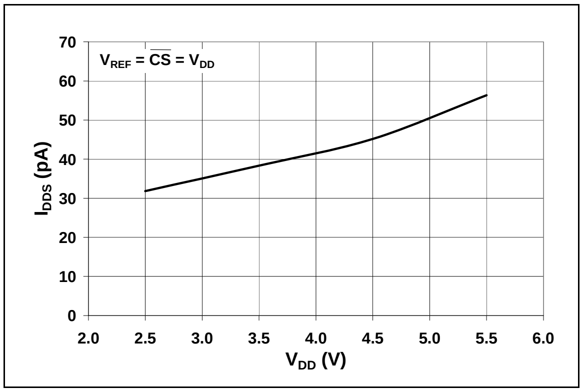


FIGURE 2-37: *I_{DDS} vs. V_{DD}.*

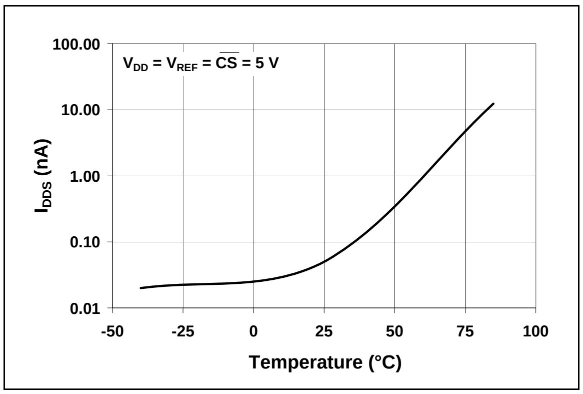


FIGURE 2-38: *I_{DDS} vs. Temperature.*

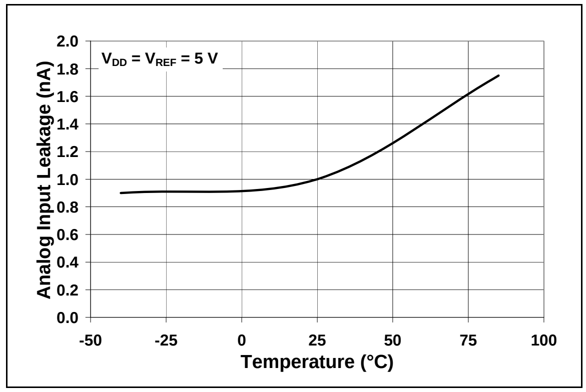


FIGURE 2-39: *Analog Input Leakage Current vs. Temperature.*

NOTES:

3.0 PIN DESCRIPTIONS

The descriptions of the pins are listed in Table 3-1. Additional descriptions of the device pins follows.

TABLE 3-1: PIN FUNCTION TABLE

MCP3004	MCP3008	Symbol	Description
PDIP, SOIC, TSSOP	PDIP, SOIC		
1	1	CH0	Analog Input
2	2	CH1	Analog Input
3	3	CH2	Analog Input
4	4	CH3	Analog Input
–	5	CH4	Analog Input
–	6	CH5	Analog Input
–	7	CH6	Analog Input
–	8	CH7	Analog Input
7	9	DGND	Digital Ground
8	10	$\overline{CS}$/SHDN	Chip Select/Shutdown Input
9	11	D_{IN}	Serial Data In
10	12	D_{OUT}	Serial Data Out
11	13	CLK	Serial Clock
12	14	AGND	Analog Ground
13	15	V_{REF}	Reference Voltage Input
14	16	V_{DD}	+2.7V to 5.5V Power Supply
5,6	–	NC	No Connection

3.1 Digital Ground (DGND)

Digital ground connection to internal digital circuitry.

3.2 Analog Ground (AGND)

Analog ground connection to internal analog circuitry.

3.3 Analog inputs (CH0 - CH7)

Analog inputs for channels 0 - 7, respectively, for the multiplexed inputs. Each pair of channels can be programmed to be used as two independent channels in single-ended mode or as a single pseudo-differential input where one channel is IN+ and one channel is IN. See **Section 4.1 “Analog Inputs”**, “Analog Inputs”, and **Section 5.0 “Serial Communication”**, “Serial Communication”, for information on programming the channel configuration.

3.4 Serial Clock (CLK)

The SPI clock pin is used to initiate a conversion and clock out each bit of the conversion as it takes place. See **Section 6.2 “Maintaining Minimum Clock Speed”**, “Maintaining Minimum Clock Speed”, for constraints on clock speed.

3.5 Serial Data Input (D_{IN})

The SPI port serial data input pin is used to load channel configuration data into the device.

3.6 Serial Data Output (D_{OUT})

The SPI serial data output pin is used to shift out the results of the A/D conversion. Data will always change on the falling edge of each clock as the conversion takes place.

3.7 Chip Select/Shutdown ($\overline{CS}$/SHDN)

The $\overline{CS}$/SHDN pin is used to initiate communication with the device when pulled low. When pulled high, it will end a conversion and put the device in low-power standby. The $\overline{CS}$/SHDN pin must be pulled high between conversions.

NOTES:

4.0 DEVICE OPERATION

The MCP3004/3008 A/D converters employ a conventional SAR architecture. With this architecture, a sample is acquired on an internal sample/hold capacitor for 1.5 clock cycles starting on the first rising edge of the serial clock once $\overline{CS}$ has been pulled low. Following this sample time, the device uses the collected charge on the internal sample and hold capacitor to produce a serial 10-bit digital output code. Conversion rates of 100 ksps are possible on the MCP3004/3008. See **Section 6.2 "Maintaining Minimum Clock Speed"**, "Maintaining Minimum Clock Speed", for information on minimum clock rates. Communication with the device is accomplished using a 4-wire SPI-compatible interface.

4.1 Analog Inputs

The MCP3004/3008 devices offer the choice of using the analog input channels configured as single-ended inputs or pseudo-differential pairs. The MCP3004 can be configured to provide two pseudo-differential input pairs or four single-ended inputs. The MCP3008 can be configured to provide four pseudo-differential input pairs or eight single-ended inputs. Configuration is done as part of the serial command before each conversion begins. When used in the pseudo-differential mode, each channel pair (i.e., CH0 and CH1, CH2 and CH3 etc.) are programmed as the IN+ and IN- inputs as part of the command string transmitted to the device. The IN+ input can range from IN- to (V_{REF} + IN-). The IN- input is limited to ±100 mV from the V_{SS} rail. The IN- input can be used to cancel small signal common-mode noise, which is present on both the IN+ and IN- inputs.

When operating in the pseudo-differential mode, if the voltage level of IN+ is equal to or less than IN-, the resultant code will be `000h`. If the voltage at IN+ is equal to or greater than {[V_{REF} + (IN-)] - 1 LSB}, then the output code will be `3FFh`. If the voltage level at IN- is more than 1 LSB below V_{SS}, the voltage level at the IN+ input will have to go below V_{SS} to see the `000h` output code. Conversely, if IN- is more than 1 LSB above V_{SS}, the `3FFh` code will not be seen unless the IN+ input level goes above V_{REF} level.

For the A/D converter to meet specification, the charge holding capacitor (C_{SAMPLE}) must be given enough time to acquire a 10-bit accurate voltage level during the 1.5 clock cycle sampling period. The analog input model is shown in Figure 4-1.

This diagram illustrates that the source impedance (R_S) adds to the internal sampling switch (R_{SS}) impedance, directly affecting the time that is required to charge the capacitor (C_{SAMPLE}). Consequently, larger source impedances increase the offset, gain and integral linearity errors of the conversion (see Figure 4-2).

4.2 Reference Input

For each device in the family, the reference input (V_{REF}) determines the analog input voltage range. As the reference input is reduced, the LSB size is reduced accordingly.

EQUATION 4-1: LSB SIZE CALCULATION

$$LSB\ Size = \frac{V_{REF}}{1024}$$

The theoretical digital output code produced by the A/D converter is a function of the analog input signal and the reference input, as shown below.

EQUATION 4-2: DIGITAL OUTPUT CODE CALCULATION

$$Digital\ Output\ Code = \frac{1024 \times V_{IN}}{V_{REF}}$$

Where:

V_{IN} = analog input voltage

V_{REF} = analog input voltage

When using an external voltage reference device, the system designer should always refer to the manufacturer's recommendations for circuit layout. Any instability in the operation of the reference device will have a direct effect on the operation of the A/D converter.

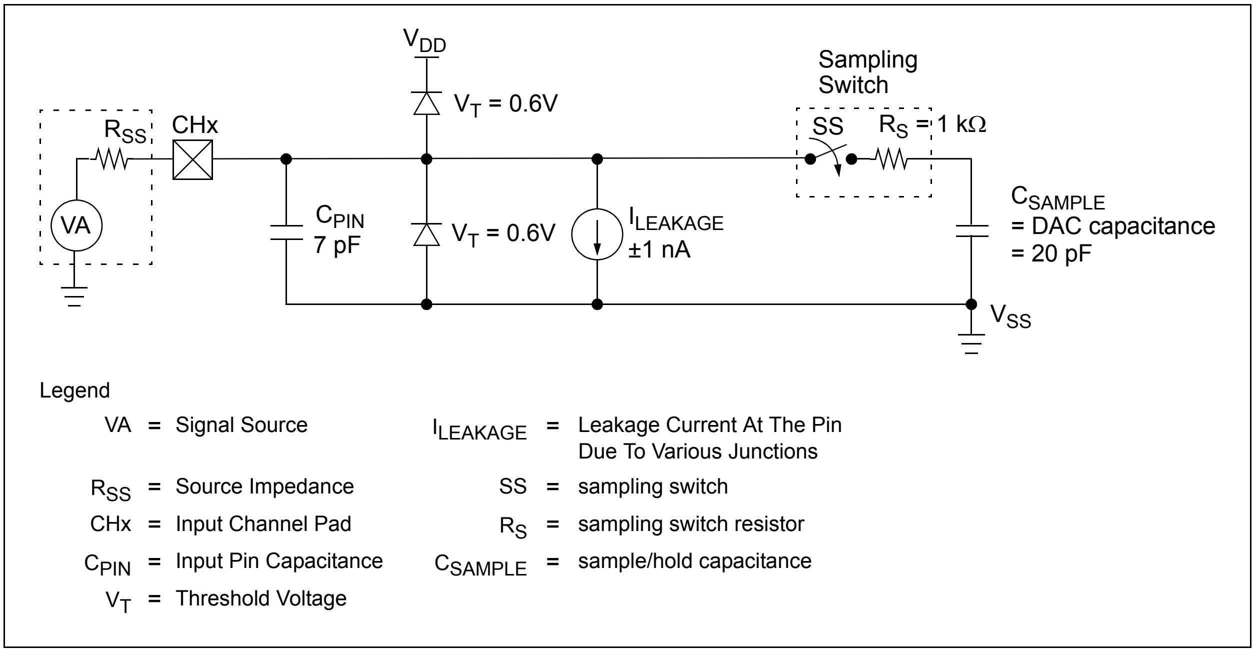


FIGURE 4-1: *Analog Input Model.*

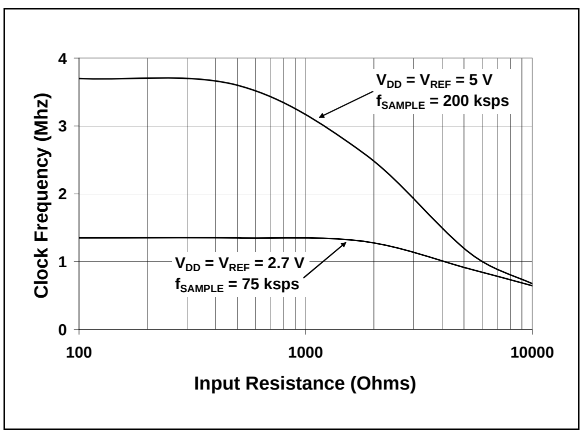


FIGURE 4-2: *Maximum Clock Frequency vs. Input resistance (R_S) to maintain less than a 0.1 LSB deviation in INL from nominal conditions.*

5.0 SERIAL COMMUNICATION

Communication with the MCP3004/3008 devices is accomplished using a standard SPI-compatible serial interface. Initiating communication with either device is done by bringing the $\overline{CS}$ line low (see Figure 5-1). If the device was powered up with the $\overline{CS}$ pin low, it must be brought high and back low to initiate communication. The first clock received with $\overline{CS}$ low and D_{IN} high will constitute a start bit. The SGL/$\overline{DIFF}$ bit follows the start bit and will determine if the conversion will be done using single-ended or differential input mode. The next three bits (D0, D1 and D2) are used to select the input channel configuration. Table 5-1 and Table 5-2 show the configuration bits for the MCP3004 and MCP3008, respectively. The device will begin to sample the analog input on the fourth rising edge of the clock after the start bit has been received. The sample period will end on the falling edge of the fifth clock following the start bit.

Once the D0 bit is input, one more clock is required to complete the sample and hold period (D_{IN} is a “don’t care” for this clock). On the falling edge of the next clock, the device will output a low null bit. The next 10 clocks will output the result of the conversion with MSB first, as shown in Figure 5-1. Data is always output from the device on the falling edge of the clock. If all 10 data bits have been transmitted and the device continues to receive clocks while the $\overline{CS}$ is held low, the device will output the conversion result LSB first, as is shown in Figure 5-2. If more clocks are provided to the device while $\overline{CS}$ is still low (after the LSB first data has been transmitted), the device will clock out zeros indefinitely.

If necessary, it is possible to bring $\overline{CS}$ low and clock in leading zeros on the D_{IN} line before the start bit. This is often done when dealing with microcontroller-based SPI ports that must send 8 bits at a time. Refer to **Section 6.1 “Using the MCP3004/3008 with Microcontroller (MCU) SPI Ports”**, “Using the MCP3004/3008 with Microcontroller (MCU) SPI Ports”, for more details on using the MCP3004/3008 devices with hardware SPI ports.

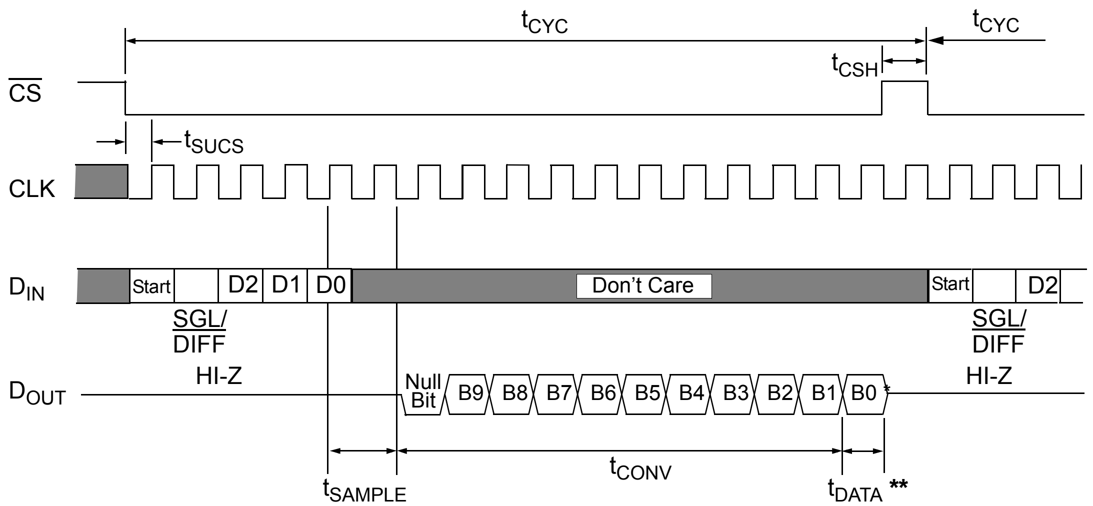
TABLE 5-1: CONFIGURE BITS FOR THE MCP3004

Control Bit Selections				Input Configuration	Channel Selection
Single/Diff	D2*	D1	D0		
1	X	0	0	single-ended	CH0
1	X	0	1	single-ended	CH1
1	X	1	0	single-ended	CH2
1	X	1	1	single-ended	CH3
0	X	0	0	differential	CH0 = IN+ CH1 = IN-
0	X	0	1	differential	CH0 = IN- CH1 = IN+
0	X	1	0	differential	CH2 = IN+ CH3 = IN-
0	X	1	1	differential	CH2 = IN- CH3 = IN+

* D2 is “don’t care” for MCP3004

TABLE 5-2: CONFIGURE BITS FOR THE MCP3008

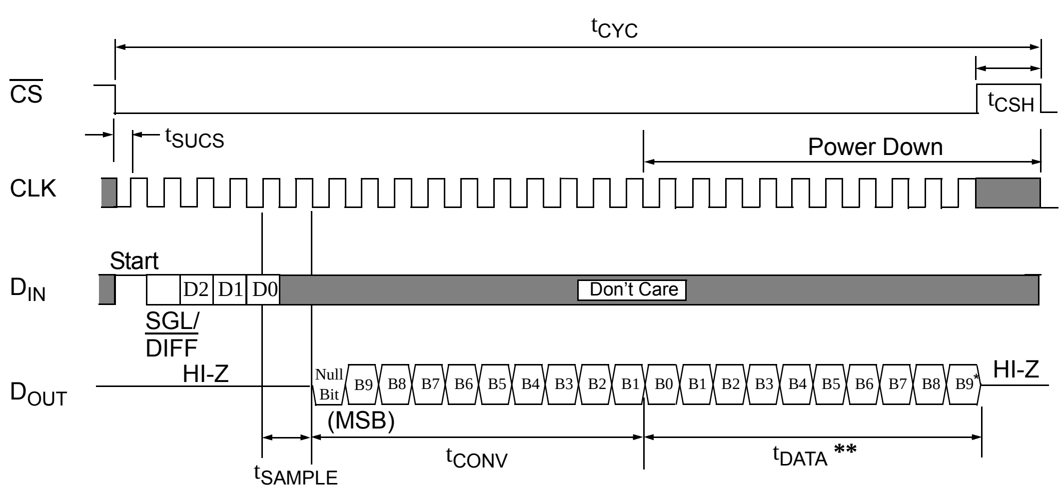
Control Bit Selections				Input Configuration	Channel Selection
Single/Diff	D2	D1	D0		
1	0	0	0	single-ended	CH0
1	0	0	1	single-ended	CH1
1	0	1	0	single-ended	CH2
1	0	1	1	single-ended	CH3
1	1	0	0	single-ended	CH4
1	1	0	1	single-ended	CH5
1	1	1	0	single-ended	CH6
1	1	1	1	single-ended	CH7
0	0	0	0	differential	CH0 = IN+ CH1 = IN-
0	0	0	1	differential	CH0 = IN- CH1 = IN+
0	0	1	0	differential	CH2 = IN+ CH3 = IN-
0	0	1	1	differential	CH2 = IN- CH3 = IN+
0	1	0	0	differential	CH4 = IN+ CH5 = IN-
0	1	0	1	differential	CH4 = IN- CH5 = IN+
0	1	1	0	differential	CH6 = IN+ CH7 = IN-
0	1	1	1	differential	CH6 = IN- CH7 = IN+



* After completing the data transfer, if further clocks are applied with $\overline{CS}$ low, the A/D converter will output LSB first data, then followed with zeros indefinitely. See Figure 5-2 below.

** t_{DATA}: during this time, the bias current and the comparator powers down while the reference input becomes a high-impedance node.

FIGURE 5-1: *Communication with the MCP3004 or MCP3008.*



* After completing the data transfer, if further clocks are applied with $\overline{CS}$ low, the A/D converter will output zeros indefinitely.

** t_{DATA}: During this time, the bias circuit and the comparator powers down while the reference input becomes a high-impedance node, leaving the CLK running to clock out LSB first data or zeroes.

FIGURE 5-2: *Communication with MCP3004 or MCP3008 in LSB First Format.*

6.0 APPLICATIONS INFORMATION

6.1 Using the MCP3004/3008 with Microcontroller (MCU) SPI Ports

With most microcontroller SPI ports, it is required to send groups of eight bits. It is also required that the microcontroller SPI port be configured to clock out data on the falling edge of clock and latch data in on the rising edge. Because communication with the MCP3004/3008 devices may not need multiples of eight clocks, it will be necessary to provide more clocks than are required. This is usually done by sending 'leading zeros' before the start bit. As an example, Figure 6-1 and Figure 6-2 shows how the MCP3004/3008 can be interfaced to a MCU with a hardware SPI port. Figure 6-1 depicts the operation shown in SPI Mode 0,0, which requires that the SCLK from the MCU idles in the 'low' state, while Figure 6-2 shows the similar case of SPI Mode 1,1, where the clock idles in the 'high' state.

As is shown in Figure 6-1, the first byte transmitted to the A/D converter contains seven leading zeros before the start bit. Arranging the leading zeros this way induces the 10 data bits to fall in positions easily manipulated by the MCU. The MSB is clocked out of the A/D converter on the falling edge of clock number 14. Once the second eight clocks have been sent to the device, the MCU receive buffer will contain five unknown bits (the output is at high-impedance for the first two clocks), the null bit and the highest order 2 bits of the conversion. Once the third byte has been sent to the device, the receive register will contain the lowest order eight bits of the conversion results. Employing this method ensures simpler manipulation of the converted data.

Figure 6-2 shows the same thing in SPI Mode 1,1, which requires that the clock idles in the high state. As with mode 0,0, the A/D converter outputs data on the falling edge of the clock and the MCU latches data from the A/D converter in on the rising edge of the clock.

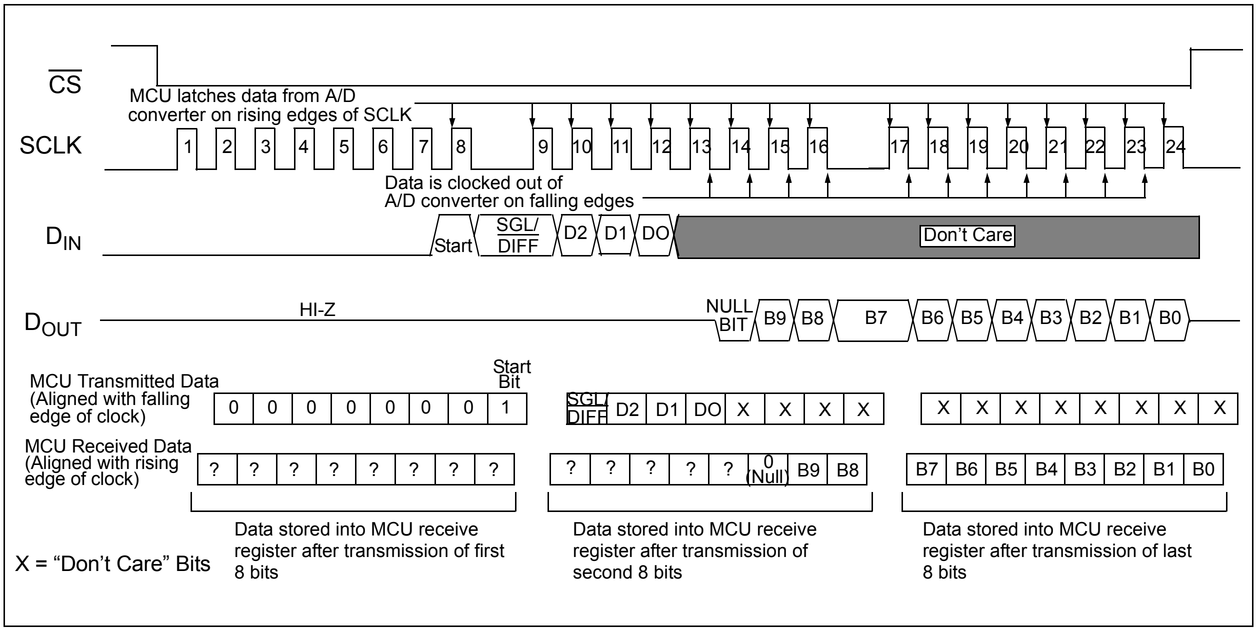


FIGURE 6-1: *SPI Communication with the MCP3004/3008 using 8-bit segments (Mode 0,0: SCLK idles low).*

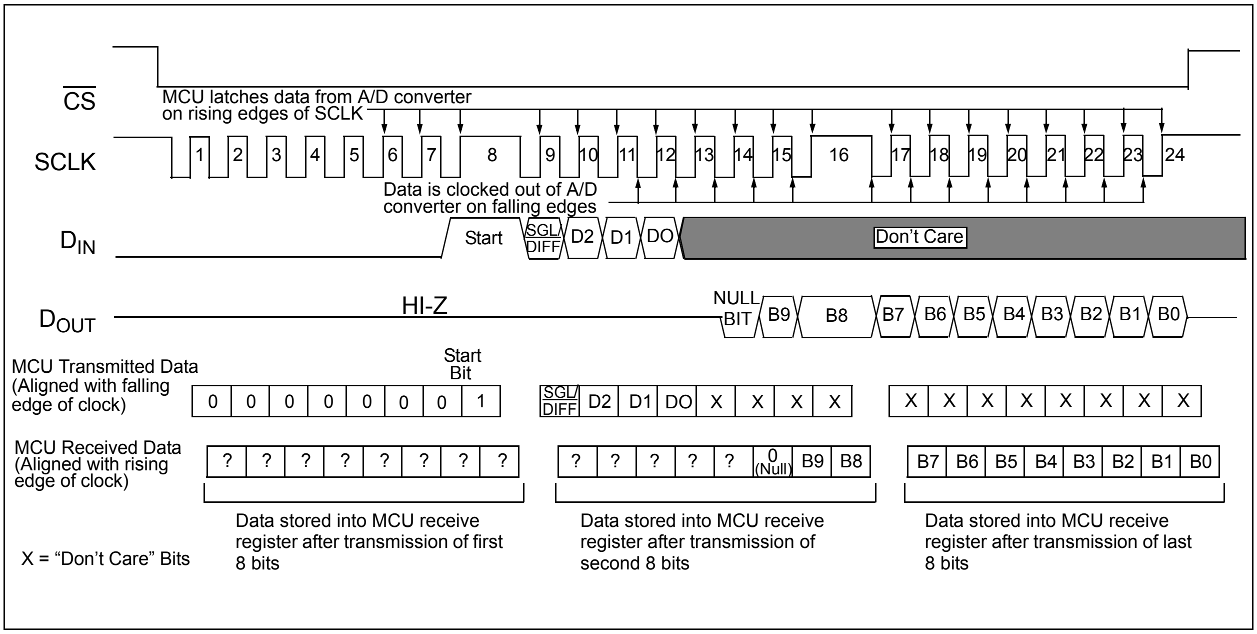


FIGURE 6-2: *SPI Communication with the MCP3004/3008 using 8-bit segments (Mode 1,1: SCLK idles high).*

6.2 Maintaining Minimum Clock Speed

When the MCP3004/3008 initiates the sample period, charge is stored on the sample capacitor. When the sample period is complete, the device converts one bit for each clock that is received. It is important for the user to note that a slow clock rate will allow charge to bleed off the sample capacitor while the conversion is taking place. At 85°C (worst case condition), the part will maintain proper charge on the sample capacitor for at least 1.2 ms after the sample period has ended. This means that the time between the end of the sample period and the time that all 10 data bits have been clocked out must not exceed 1.2 ms (effective clock frequency of 10 kHz). Failure to meet this criterion may introduce linearity errors into the conversion outside the rated specifications. It should be noted that during the entire conversion cycle, the A/D converter does not require a constant clock speed or duty cycle, as long as all timing specifications are met.

6.3 Buffering/Filtering the Analog Inputs

If the signal source for the A/D converter is not a low-impedance source, it will have to be buffered or inaccurate conversion results may occur (see Figure 4-2). It is also recommended that a filter be used to eliminate any signals that may be aliased back in to the conversion results, as is illustrated in Figure 6-3, where an op amp is used to drive, filter and gain the analog input of the MCP3004/3008. This amplifier provides a low-impedance source for the converter input, plus a low-pass filter, which eliminates unwanted high-frequency noise.

Low-pass (anti-aliasing) filters can be designed using Microchip's free interactive FilterLab® software. FilterLab will calculate capacitor and resistors values, as well as determine the number of poles that are required for the application. For more information on filtering signals, see AN699, *"Anti-Aliasing Analog Filters for Data Acquisition Systems"*.

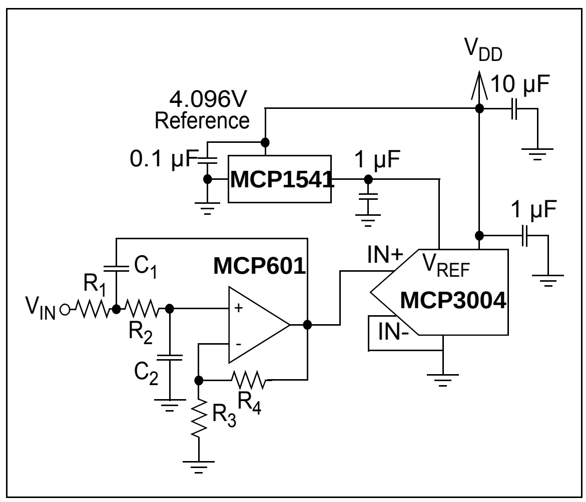


FIGURE 6-3: *The MCP601 Operational Amplifier is used to implement a second order anti-aliasing filter for the signal being converted by the MCP3004.*

6.4 Layout Considerations

When laying out a printed circuit board for use with analog components, care should be taken to reduce noise wherever possible. A bypass capacitor should always be used with this device and should be placed as close as possible to the device pin. A bypass capacitor value of 1 µF is recommended.

Digital and analog traces should be separated as much as possible on the board, with no traces running underneath the device or bypass capacitor. Extra precautions should be taken to keep traces with high-frequency signals (such as clock lines) as far as possible from analog traces.

Use of an analog ground plane is recommended in order to keep the ground potential the same for all devices on the board. Providing V_{DD} connections to devices in a "star" configuration can also reduce noise by eliminating return current paths and associated errors (see Figure 6-4). For more information on layout tips when using A/D converters, refer to AN688, *"Layout Tips for 12-Bit A/D Converter Applications"*.

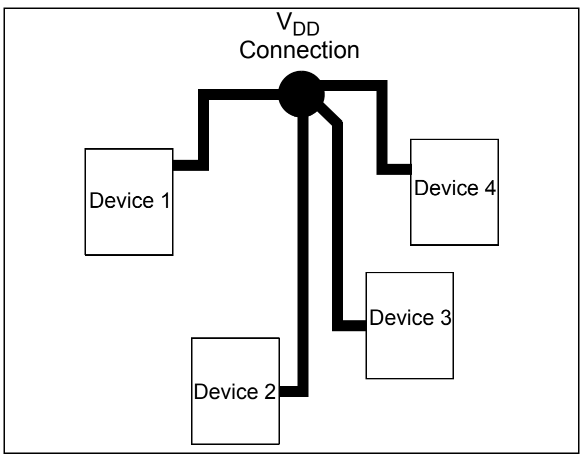


FIGURE 6-4: *V_{DD} traces arranged in a 'Star' configuration in order to reduce errors caused by current return paths.*

6.5 Utilizing the Digital and Analog Ground Pins

The MCP3004/3008 devices provide both digital and analog ground connections to provide additional means of noise reduction. As is shown in Figure 6-5, the analog and digital circuitry is separated internal to the device. This reduces noise from the digital portion of the device being coupled into the analog portion of the device. The two grounds are connected internally through the substrate which has a resistance of 5 -10Ω.

If no ground plane is utilized, both grounds must be connected to V_{SS} on the board. If a ground plane is available, both digital and analog ground pins should be connected to the analog ground plane. If both an analog and a digital ground plane are available, both the digital and the analog ground pins should be connected to the analog ground plane. Following these steps will reduce the amount of digital noise from the rest of the board being coupled into the A/D converter.

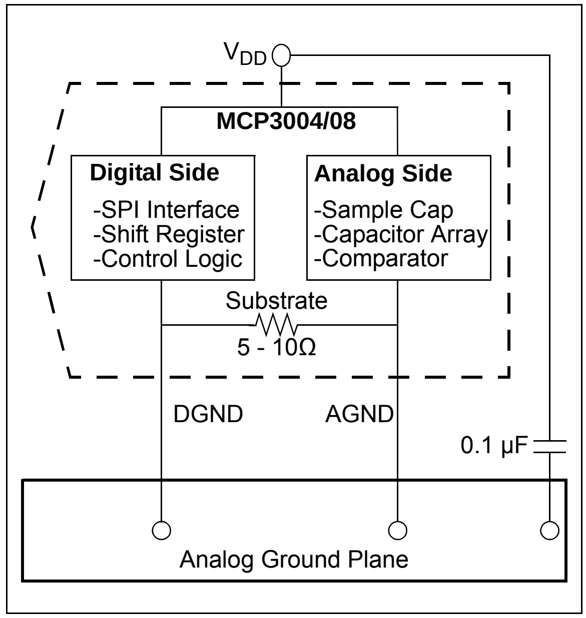


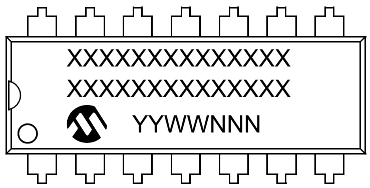
FIGURE 6-5: *Separation of Analog and Digital Ground Pins.*

NOTES:

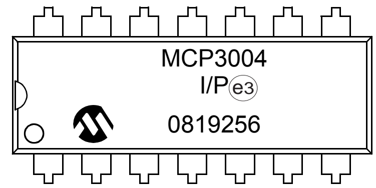
7.0 PACKAGING INFORMATION

7.1 Package Marking Information

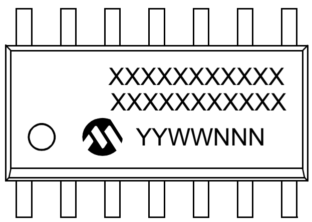
14-Lead PDIP (300 mil)



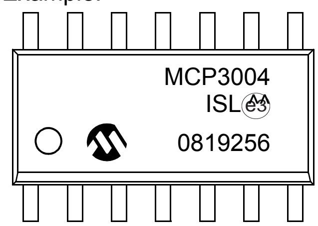
Example:



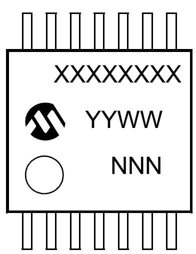
14-Lead SOIC (150 mil)



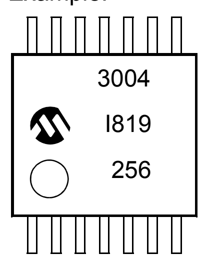
Example:



14-Lead TSSOP (4.4mm) *



Example:

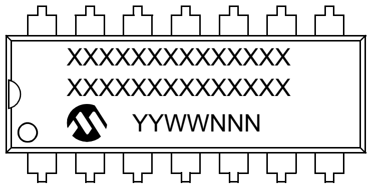


Legend:	XX...X	Customer-specific information
	Y	Year code (last digit of calendar year)
	YY	Year code (last 2 digits of calendar year)
	WW	Week code (week of January 1 is week '01')
	NNN	Alphanumeric traceability code
	e3	Pb-free JEDEC designator for Matte Tin (Sn)
	*	This package is Pb-free. The Pb-free JEDEC designator (e3) can be found on the outer packaging for this package.

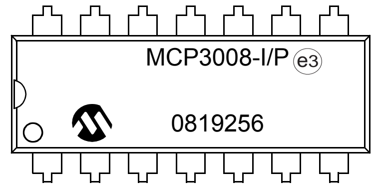
Note: In the event the full Microchip part number cannot be marked on one line, it will be carried over to the next line, thus limiting the number of available characters for customer-specific information.

Package Marking Information (Continued)

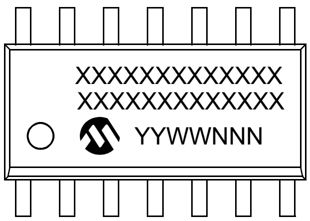
16-Lead PDIP (300 mil) (**MCP3008**)



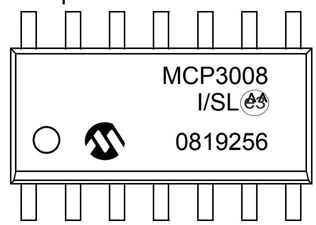
Example:



16-Lead SOIC (150 mil) (**MCP3008**)

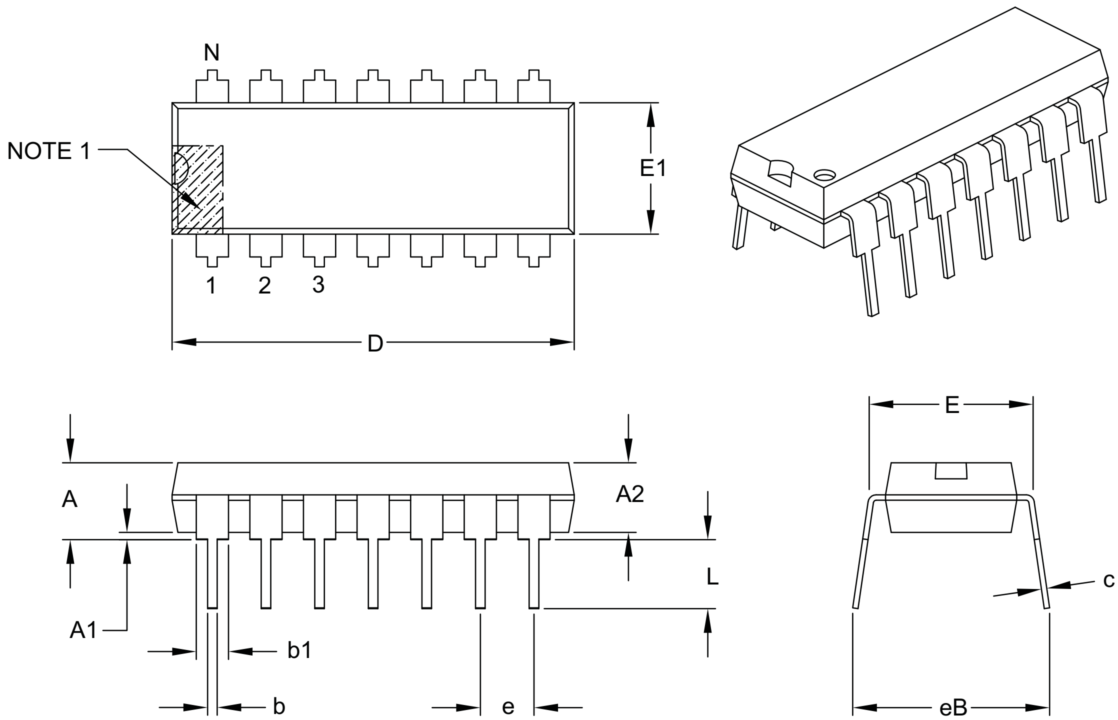


Example:



14-Lead Plastic Dual In-Line (P) – 300 mil Body [PDIP]

Note:	For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



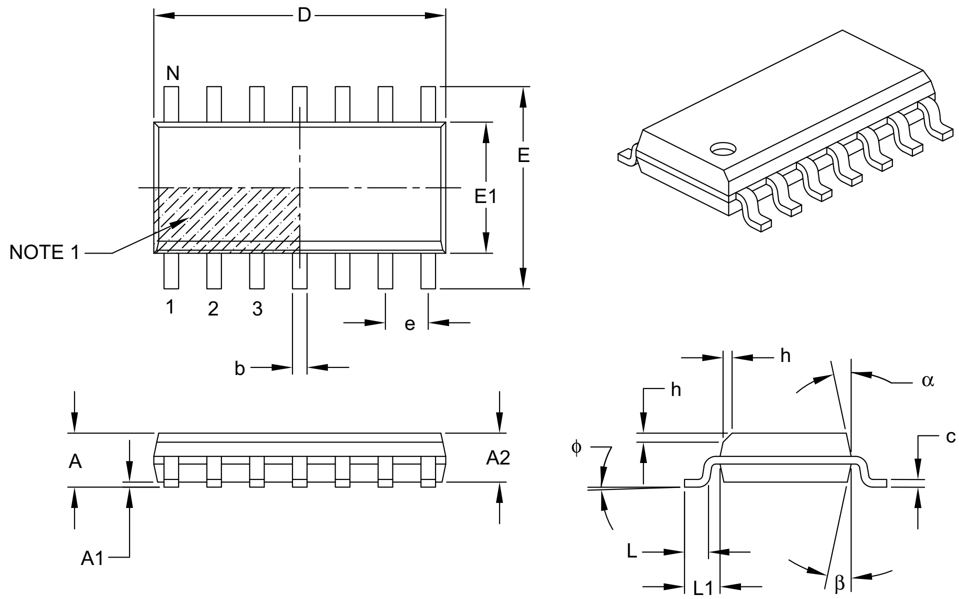
		Units	INCHES	
		Dimension Limits	MIN	NOM	MAX
Number of Pins	N	14		
Pitch	e	.100 BSC		
Top to Seating Plane	A	–	–	.210
Molded Package Thickness	A2	.115	.130	.195
Base to Seating Plane	A1	.015	–	–
Shoulder to Shoulder Width	E	.290	.310	.325
Molded Package Width	E1	.240	.250	.280
Overall Length	D	.735	.750	.775
Tip to Seating Plane	L	.115	.130	.150
Lead Thickness	c	.008	.010	.015
Upper Lead Width	b1	.045	.060	.070
Lower Lead Width	b	.014	.018	.022
Overall Row Spacing §	eB	–	–	.430

Notes:
1. Pin 1 visual index feature may vary, but must be located with the hatched area.
2. § Significant Characteristic.
3. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" per side.
4. Dimensioning and tolerancing per ASME Y14.5M.
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing C04-005B

14-Lead Plastic Small Outline (SL) – Narrow, 3.90 mm Body [SOIC]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



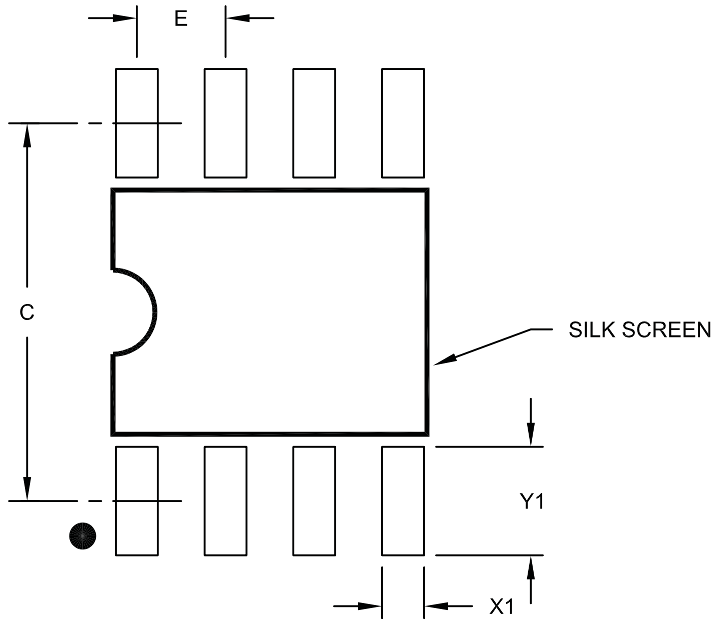
		Units	MILLIMETERS	
		Dimension Limits	MIN	NOM	MAX
Number of Pins	N		14	
Pitch	e		1.27 BSC	
Overall Height	A	–	–	1.75
Molded Package Thickness	A2	1.25	–	–
Standoff §	A1	0.10	–	0.25
Overall Width	E		6.00 BSC	
Molded Package Width	E1		3.90 BSC	
Overall Length	D		8.65 BSC	
Chamfer (optional)	h	0.25	–	0.50
Foot Length	L	0.40	–	1.27
Footprint	L1		1.04 REF	
Foot Angle	ϕ	0°	–	8°
Lead Thickness	c	0.17	–	0.25
Lead Width	b	0.31	–	0.51
Mold Draft Angle Top	α	5°	–	15°
Mold Draft Angle Bottom	β	5°	–	15°

Notes:
1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. § Significant Characteristic.
3. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.15 mm per side.
4. Dimensioning and tolerancing per ASME Y14.5M.
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-065B

8-Lead Plastic Small Outline (SN) – Narrow, 3.90 mm Body [SOIC]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



RECOMMENDED LAND PATTERN

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E	1.27 BSC		
Contact Pad Spacing	C		5.40	
Contact Pad Width (X8)	X1			0.60
Contact Pad Length (X8)	Y1			1.55

Notes:

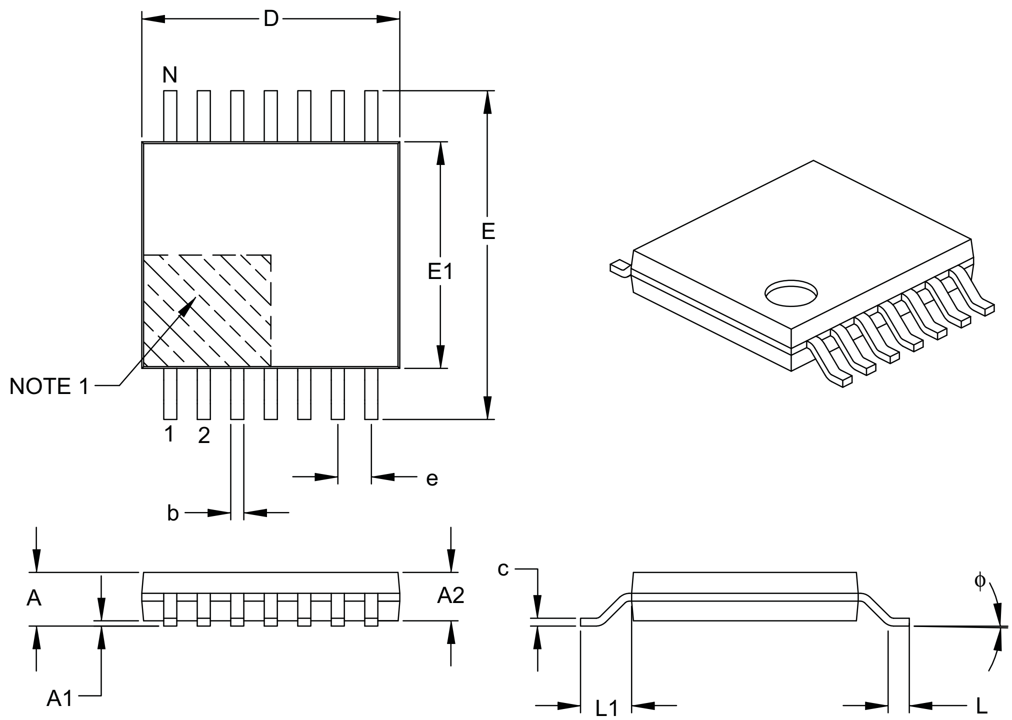
1. Dimensioning and tolerancing per ASME Y14.5M

   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2057A

14-Lead Plastic Thin Shrink Small Outline (ST) – 4.4 mm Body [TSSOP]

Note:	For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Pins	N		14	
Pitch	e		0.65 BSC	
Overall Height	A	–	–	1.20
Molded Package Thickness	A2	0.80	1.00	1.05
Standoff	A1	0.05	–	0.15
Overall Width	E		6.40 BSC	
Molded Package Width	E1	4.30	4.40	4.50
Molded Package Length	D	4.90	5.00	5.10
Foot Length	L	0.45	0.60	0.75
Footprint	L1		1.00 REF	
Foot Angle	ϕ	0°	–	8°
Lead Thickness	c	0.09	–	0.20
Lead Width	b	0.19	–	0.30

Notes:
1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.15 mm per side.
3. Dimensioning and tolerancing per ASME Y14.5M.

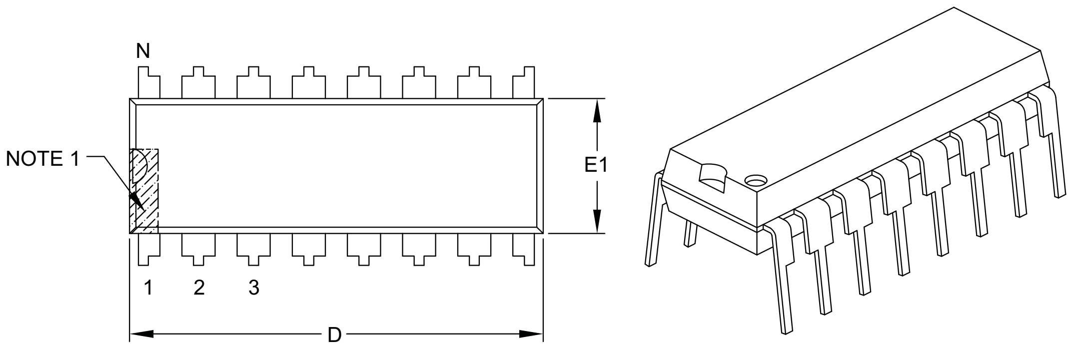
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

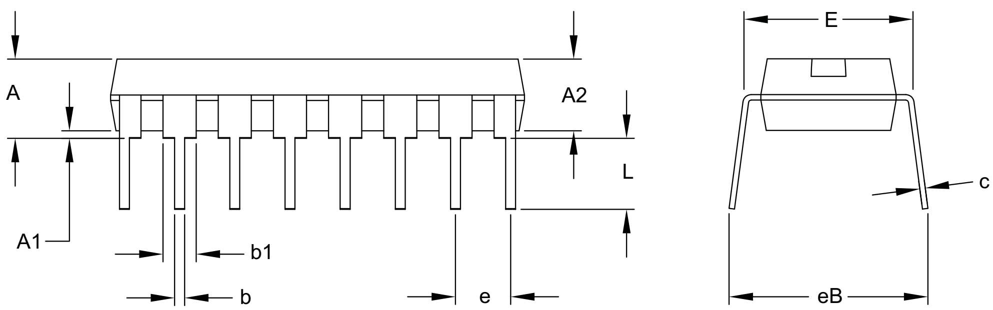
   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-087B

16-Lead Plastic Dual In-Line (P) – 300 mil Body [PDIP]

Note:	For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging





Units		INCHES		
Dimension Limits		MIN	NOM	MAX
Number of Pins	N	16		
Pitch	e	.100 BSC		
Top to Seating Plane	A	–	–	.210
Molded Package Thickness	A2	.115	.130	.195
Base to Seating Plane	A1	.015	–	–
Shoulder to Shoulder Width	E	.290	.310	.325
Molded Package Width	E1	.240	.250	.280
Overall Length	D	.735	.755	.775
Tip to Seating Plane	L	.115	.130	.150
Lead Thickness	c	.008	.010	.015
Upper Lead Width	b1	.045	.060	.070
Lower Lead Width	b	.014	.018	.022
Overall Row Spacing §	eB	–	–	.430

Notes:

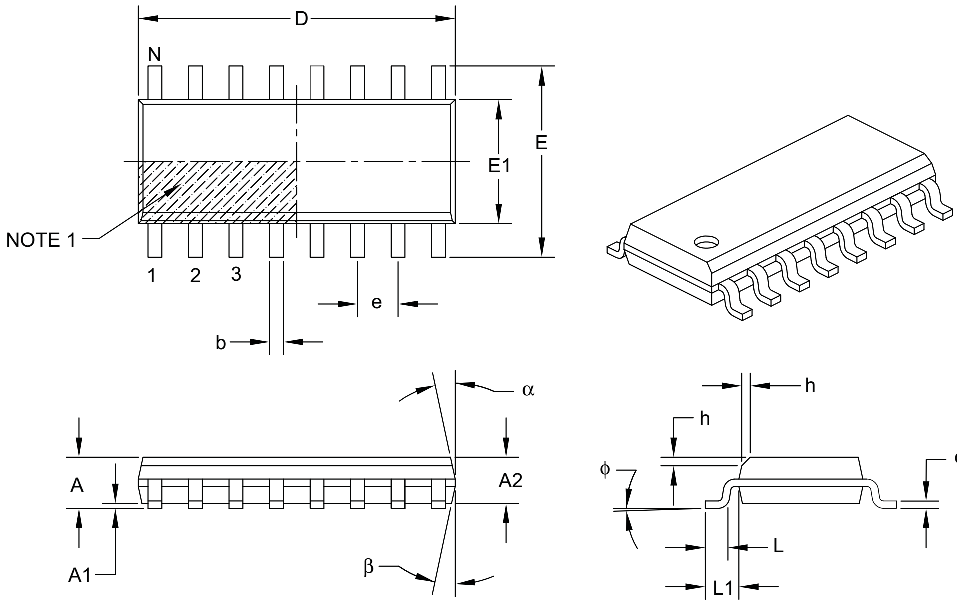
1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. § Significant Characteristic.
3. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" per side.
4. Dimensioning and tolerancing per ASME Y14.5M.

   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing C04-017B

16-Lead Plastic Small Outline (SL) – Narrow, 3.90 mm Body [SOIC]

Note:	For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



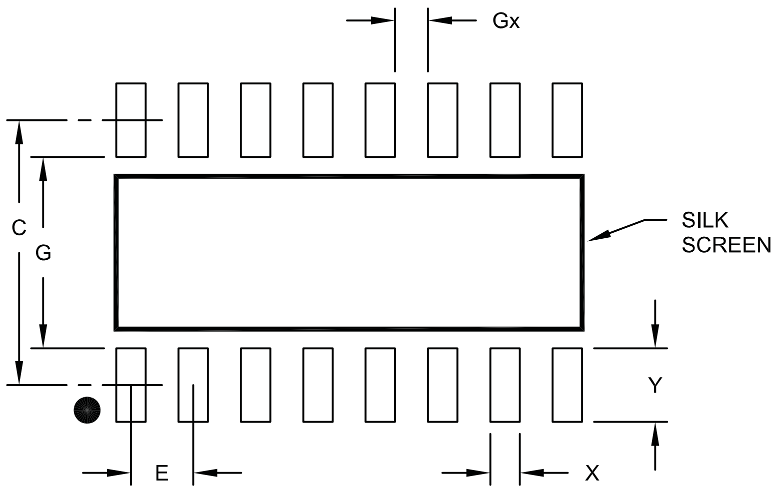
	Units	MILLIMETERS		
	Dimension Limits	MIN	NOM	MAX
Number of Pins	N	16		
Pitch	e	1.27 BSC		
Overall Height	A	–	–	1.75
Molded Package Thickness	A2	1.25	–	–
Standoff §	A1	0.10	–	0.25
Overall Width	E	6.00 BSC		
Molded Package Width	E1	3.90 BSC		
Overall Length	D	9.90 BSC		
Chamfer (optional)	h	0.25	–	0.50
Foot Length	L	0.40	–	1.27
Footprint	L1	1.04 REF		
Foot Angle	φ	0°	–	8°
Lead Thickness	c	0.17	–	0.25
Lead Width	b	0.31	–	0.51
Mold Draft Angle Top	α	5°	–	15°
Mold Draft Angle Bottom	β	5°	–	15°

Notes:
1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. § Significant Characteristic.
3. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.15 mm per side.
4. Dimensioning and tolerancing per ASME Y14.5M.
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-108B

16-Lead Plastic Small Outline (SL) - Narrow, 3.90 mm Body [SOIC]

Note:	For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



RECOMMENDED LAND PATTERN

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E	1.27 BSC		
Contact Pad Spacing	C		5.40	
Contact Pad Width	X			0.60
Contact Pad Length	Y			1.50
Distance Between Pads	Gx	0.67		
Distance Between Pads	G	3.90		

Notes:

1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2108A

NOTES:

APPENDIX A: REVISION HISTORY

Revision D (December 2008)

The following is the list of modifications:

1. Updates to **Section 7.0 "Packaging Information"**.

Revision C (January 2007)

The following is the list of modifications:

1. Updates to the packaging diagrams.

Revision B (May 2002)

The following is the list of modifications:

1. Undocumented changes.

Revision A (February 2000)

- Initial release of this document.

NOTES:

PRODUCT IDENTIFICATION SYSTEM

To order or obtain information, e.g., on pricing or delivery, refer to the factory or the listed sales office.

PART NO.	X	/XX
Device	Temperature Range	Package

Device	MCP3004: 4-Channel 10-Bit Serial A/D Converter MCP3004T: 4-Channel 10-Bit Serial A/D Converter (Tape and Reel) MCP3008: 8-Channel 10-Bit Serial A/D Converter MCP3008T: 8-Channel 10-Bit Serial A/D Converter (Tape and Reel)
Temperature Range	I = -40°C to +85°C (Industrial)
Package	P = Plastic DIP (300 mil Body), 14-lead, 16-lead SL = Plastic SOIC (150 mil Body), 14-lead, 16-lead ST = Plastic TSSOP (4.4mm), 14-lead

Examples:

a) MCP3004-I/P: Industrial Temperature, PDIP package.
b) MCP3004-I/SL: Industrial Temperature, SOIC package.
c) MCP3004-I/ST: Industrial Temperature, TSSOP package.
d) MCP3004T-I/ST: Industrial Temperature, TSSOP package, Tape and Reel.

a) MCP3008-I/P: Industrial Temperature, PDIP package.
b) MCP3008-I/SL: Industrial Temperature, SOIC package.

NOTES:

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as "unbreakable."

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

QUALITY MANAGEMENT SYSTEM
CERTIFIED BY DNV
ISO/TS 16949:2002

Microchip received ISO/TS-16949:2002 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California and India. The Company's quality system processes and procedures are for its PIC® MCUs and dsPIC® DSCs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.

WORLDWIDE SALES AND SERVICE

AMERICAS

Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200
Fax: 480-792-7277
Technical Support:
http://support.microchip.com
Web Address:
www.microchip.com

Atlanta
Duluth, GA
Tel: 678-957-9614
Fax: 678-957-1455

Boston
Westborough, MA
Tel: 774-760-0087
Fax: 774-760-0088

Chicago
Itasca, IL
Tel: 630-285-0071
Fax: 630-285-0075

Dallas
Addison, TX
Tel: 972-818-7423
Fax: 972-818-2924

Detroit
Farmington Hills, MI
Tel: 248-538-2250
Fax: 248-538-2260

Kokomo
Kokomo, IN
Tel: 765-864-8360
Fax: 765-864-8387

Los Angeles
Mission Viejo, CA
Tel: 949-462-9523
Fax: 949-462-9608

Santa Clara
Santa Clara, CA
Tel: 408-961-6444
Fax: 408-961-6445

Toronto
Mississauga, Ontario,
Canada
Tel: 905-673-0699
Fax: 905-673-6509

ASIA/PACIFIC

Asia Pacific Office
Suites 3707-14, 37th Floor
Tower 6, The Gateway
Harbour City, Kowloon
Hong Kong
Tel: 852-2401-1200
Fax: 852-2401-3431

Australia - Sydney
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

China - Beijing
Tel: 86-10-8528-2100
Fax: 86-10-8528-2104

China - Chengdu
Tel: 86-28-8665-5511
Fax: 86-28-8665-7889

China - Hong Kong SAR
Tel: 852-2401-1200
Fax: 852-2401-3431

China - Nanjing
Tel: 86-25-8473-2460
Fax: 86-25-8473-2470

China - Qingdao
Tel: 86-532-8502-7355
Fax: 86-532-8502-7205

China - Shanghai
Tel: 86-21-5407-5533
Fax: 86-21-5407-5066

China - Shenyang
Tel: 86-24-2334-2829
Fax: 86-24-2334-2393

China - Shenzhen
Tel: 86-755-8203-2660
Fax: 86-755-8203-1760

China - Wuhan
Tel: 86-27-5980-5300
Fax: 86-27-5980-5118

China - Xiamen
Tel: 86-592-2388138
Fax: 86-592-2388130

China - Xian
Tel: 86-29-8833-7252
Fax: 86-29-8833-7256

China - Zhuhai
Tel: 86-756-3210040
Fax: 86-756-3210049

ASIA/PACIFIC

India - Bangalore
Tel: 91-80-4182-8400
Fax: 91-80-4182-8422

India - New Delhi
Tel: 91-11-4160-8631
Fax: 91-11-4160-8632

India - Pune
Tel: 91-20-2566-1512
Fax: 91-20-2566-1513

Japan - Yokohama
Tel: 81-45-471- 6166
Fax: 81-45-471-6122

Korea - Daegu
Tel: 82-53-744-4301
Fax: 82-53-744-4302

Korea - Seoul
Tel: 82-2-554-7200
Fax: 82-2-558-5932 or
82-2-558-5934

Malaysia - Kuala Lumpur
Tel: 60-3-6201-9857
Fax: 60-3-6201-9859

Malaysia - Penang
Tel: 60-4-227-8870
Fax: 60-4-227-4068

Philippines - Manila
Tel: 63-2-634-9065
Fax: 63-2-634-9069

Singapore
Tel: 65-6334-8870
Fax: 65-6334-8850

Taiwan - Hsin Chu
Tel: 886-3-572-9526
Fax: 886-3-572-6459

Taiwan - Kaohsiung
Tel: 886-7-536-4818
Fax: 886-7-536-4803

Taiwan - Taipei
Tel: 886-2-2500-6610
Fax: 886-2-2508-0102

Thailand - Bangkok
Tel: 66-2-694-1351
Fax: 66-2-694-1350

EUROPE

Austria - Wels
Tel: 43-7242-2244-39
Fax: 43-7242-2244-393

Denmark - Copenhagen
Tel: 45-4450-2828
Fax: 45-4485-2829

France - Paris
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

Germany - Munich
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

Italy - Milan
Tel: 39-0331-742611
Fax: 39-0331-466781

Netherlands - Drunen
Tel: 31-416-690399
Fax: 31-416-690340

Spain - Madrid
Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

UK - Wokingham
Tel: 44-118-921-5869
Fax: 44-118-921-5820

01/02/08